XLIV Symposium on Operational Research
SYM-OP-IS 2017
Zlatibor, 25-28 Septembar 2017

PARALELIZACIJA METAHEURISTIČKIH METODA[1]

PARALLELIZATION OF METAHEURISTIC METHODS

TATJANA DAVIDOVIĆ[1]

[1] Matematički institut SANU, tanjad@mi.sanu.ac.rs

Rezime: *Metaheuristike su stohastički algoritmi koji se koriste u optimizaciji. One predstavljaju moćne alate za rešavanje složenih optimizacionih problema. Međutim, za probleme iz realnog života ponekad nije moguće dobiti efikasno rešenje u "zadovoljavajućem" vremenu rada ovih algoritama. Paralelizacija je jedan od načina za prevazilaženje ovog problema. Cilj ovog rada je da predstavi trenutno stanje u istraživanjima, glavne ideje i strategije koje se koriste u paralelizaciji i da istakne opšte principe koji važe za sve tipove metaheuristika.*

Ključne reči: *Problemi optimizacije, Kvalitetna rešenja, Metaheuristike, Paralelno izvršavanje, Višeprocesorski sistemi.*

Abstract: *Meta-heuristics are stochastic algorithms widely used in optimization. They are known as powerful tools for dealing with hard optimization problems. However, sometimes real life instances cannot be treated efficiently in "reasonable" execution time. Parallelization is a way to deal with this problem. The objective of this paper is to present a state-of-the-art survey of the main parallelization ideas and strategies, and to discuss general design principles applicable to all meta-heuristic classes.*

Keywords: *Optimization problems, High quality solutions, Metaheuristics, Parallel execution, Multiprocessors.*

1. UVOD

Osnovni cilj paralelnog izvršavanja je da ubrza izračunavanja deleći ih između više procesora. Sa stanovišta dizajniranja algoritama, jednostavne strategije paralelizacije koriste parcijalni poredak među koracima algoritma, tj. postojanje skupa operacija koje se mogu izvršiti paralelno (istovremeno, nezavisno) bez uticaja na metodu rešavanja i na dobijeno rešenje. One koriste „prirodni" paralelizam koji već postoji u algoritmu.

Razvoj paralelnih algoritama u današnje vreme je postao veoma uobičajena istraživačka tema. Višeprocesorski sistemi najrazličitijih vrsta predstavljaju standardnu opremu naučnih i obrazovnih institucija. Svi savremeni računari imaju više procesora među svojim komponentama, bilo da su u pitanju višejezgarni procesori (multicore sistemi) ili grafičke kartice (Graphic Processing Units, GPU). Sa druge strane, akademske GRID i CLOUD mreže pružaju mogućnost korišćenja standardnih višeprocesorskih sistema (tzv. multicomputer systems) koji se sastoje od identičnih (homogenih) ili različitih (heterogenih) jednoprocesorskih računara povezanih na proizvoljan način ili u neku zadatu topologiju.

Algoritmi koji se izvršavaju nad neregularnim strukturama podataka, kao što su grafovi, ili nad podacima sa jakim zavisnostima među različitim operacijama generalno su teški su za paralelizaciju. Metaheuristike takođe spadaju u klasu algoritama koji se teško paralelizuju, ali ima interesantnih rezultata iz te oblasti i novih izazova. Iako su na mnogim primerima primene pokazale svoju efikasnost, prilikom rešavanja nekih složenijih NP-teških problema ili problema velikih mogu se pojaviti teškoće u pronalaženju kvalitetnih rešenja (Alba *et al*. 2013, Smutnicki and Bożejko 2015). Jedan od načina da se ove teškoće prevaziđu je da se originalne metaheurističke metode paralelizuju i da se njihovim distribuiranim izvršavanjem na više procesora ostvari bar jedan od mogućih ciljeva: ubrzavanje neophodnih izračunavanja (tj. dobijanje rešenja za kraće vreme izvršavanja metode) ili popravljanje kvaliteta rešenja (tj. dobijanje boljeg rešenja u istom ili kraćem vremenu izvršavanja).

U radovima Crainic and Hail (2005), Crainic and Toulouse (2010) i Alba *et al.* (2013) data je generalna slika o stanju u oblasti paralelnih metaheuristika. Prikazano je dotadašnje stanje u razvoju paralelnih

[1] Rad je finansiran od Ministarstva prosvete, nauke i tehnološkog razvoja republike Srbije preko projekata OI174033 i F159

metaheuristika, njihovoj primeni i dobijenim rezultatima, ali i diskusija uopštenih principa za dizajn i implementaciju koji se mogu primeniti na većinu metaheuristika. Opisano je i kako su ti principi primenjeni na metaheuristike koje koriste jedno rešenje i metaheuristike bazirane ne populaciji rešenja. Konkretno, razmatrani su primeri genetskih algoritama, tabu pretraživanja i simuliranog kaljenja.

U proteklih nekoliko godina intenzivno se radilo na paralelizaciji metaheurističkih metoda, pa je cilj ovog rada da izdvoji nove trendove i rezultate. Rad je organizovan u 6 odeljaka. U narednom odeljku opisana je klasifikacija strategija za paralelizaciju metaheuristika kao i karakteristike najrasprostranjenijih višeprocesorskih sistema. Odeljak 3 bavi se upoređivanjem različitih paralelnih verzija metaheuristika. Najčešće paralelne implementacije metaheuristika koje koriste jedno rešenje opisane su u odeljku 4, dok su odgovarajuće implementacije populacionih metaheuristika sumirane u odeljku 5. Zaključna razmatranja sadržana su u odeljku 6.

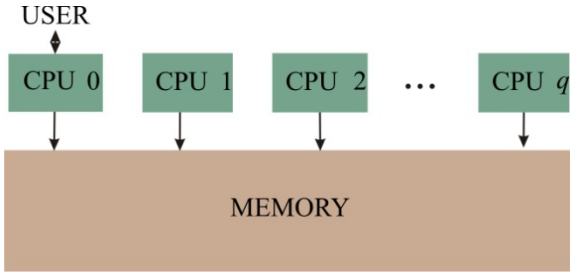
2. PARALELIZACIJA METAHEURISTIKA

Glavni cilj paralelizacije je da se ubrzaju izračunavanja potrebna da se reši određeni problem, angažovanjem više procesora i deljenjem ukupne količine posla između njih. Za stohastičke algoritme, metaheuristike u ovom slučaju, paralelizacijom je moguće postići nekoliko ciljeva (Talbi, 2009): ubrzava se pretraga (tj. skraćuje vreme za pretragu); poboljšava se kvalitet dobijenih rešenja (omogućavanjem pretraživanja kroz različite delove prostora rešenja); poboljšava se robusnost (u smislu rešavanja različitih problema optimizacije i različitih instanci datog problema na efikasan način, robusnost se takođe može meriti u smislu osetljivosti metaheuristike na vrednosti parametara); i omogućava se rešavanje instanci velikih dimenzija (tj. rešavanje primera koji su preveliki za sekvencijalne računare). Moguće je ostvariti i kombinaciju ciljeva, tj. paralelno izvršavanje može da obezbedi efikasno pretraživanje kroz različite regione prostora rešenja i tako dovede do poboljšanja kvaliteta konačnog rešenja u kratkom vremenu izvršavanja.

Na paralelizaciji metaheurističkih metoda radilo se intenzivno u proteklih dvadesetak godina (videti pregledne radove Alba *et al.* 2013 i Crainic *et al.* 2014), bilo da su u pitanju teorijski aspekti paralelizacije ili praktične primene paralelnih metaheuristika na različite probleme optimizacije. Jedan od prvih radova koji uvodi klasifikaciju strategija paralelizacije je Verhoven and Aarts (1995). Ova klasifikacija, zasnovana je na kontroli procesa pretrage i razlikuje dve osnovne grupe strategija: centralno kontrolisane (Single walk) i distribuirane (Multiple walk). Detaljnija klasifikacija, koja uzima u obzir komunikacione aspekte i parametre pretrage, predložena je u Crainic and Hail (2005), a u ovom radu biće ukratko opisana.

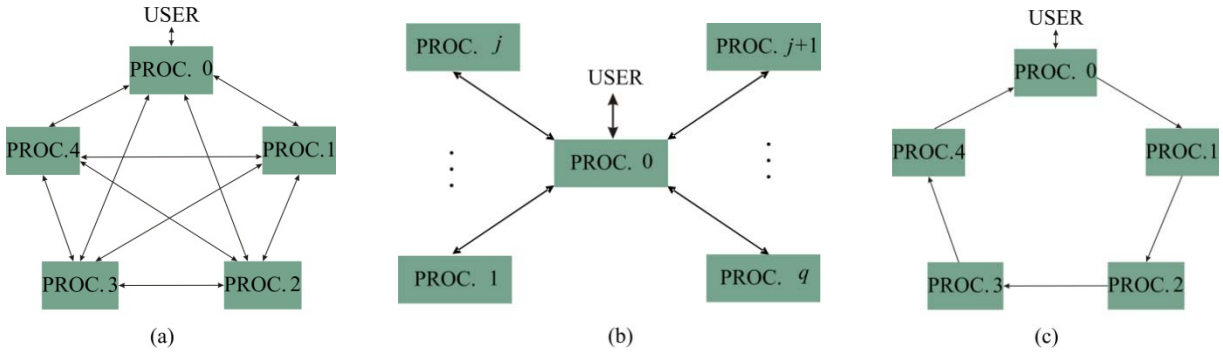
Kao što je već rečeno, klasifikacija strategija za paralelizaciju metaheurističkih metoda predložena u Crainic and Hail (2005) razmatra tri glavna aspekta paralelnog izvršavanja: kontrolu pretrage, komunikacije i parametre pretrage. Takav pristup rezultirao je 3D-taksonomijom sa poljima $\mathcal{X}/\mathcal{Y}/\mathcal{Z}$. Polje $\mathcal{X}$ koristi se da označi tip kontrole pretrage, koja može biti centralizovana (u daljem tekstu: 1C) ili distribuirana (u daljem tekstu: pC). $\mathcal{Y}$ opisuje dva aspekta u komunikaciji između paralelnih procesa: sinhronizaciju i vrste podataka koji se razmenjuju. Postoje četiri mogućnosti za vrednost polja $\mathcal{Y}$: stroga sinhrona komunikacija (u daljem tekstu: RS), sinhrona sa modifikovanjem prenesenih podataka (u daljem tekstu: KS), asinhrona (u daljem tekstu: C) i asinhrona sa učenjem iz prenetih podataka (u daljem tekstu: KC). Polje $\mathcal{Z}$ determiniše deo pretrage koji izvršava svaki od paralelnih procesa na osnovu početne tačke i metode pretrage. Svaki proces može da počne od iste ili različite početne tačke, a pri tome procesi mogu da koriste istu ili različite metode pretraživanja. Zbog toga, postoje četiri kombinacije vrednosti za $\mathcal{Z}$: ista početna tačka-ista metoda pretrage (u daljem tekstu: SPSS), ista početna tačka-drugačije metode (u daljem tekstu: SPDS), različite početne tačke-ista metoda pretrage (u daljem tekstu: MPSS), različite početne tačke-različite metode pretraživanja (u daljem tekstu: MPDS). Ove vrednosti dovoljne su da opišu bilo koju strategiju paralelizacije, ali konkretna implementacija svake od opisanih strategija može da varira, zavisno od karakteristika višeprocesorskog sistema na kome će se paralelna verzija algoritma izvršavati kao i problema koji se rešava.

Kada je reč o ciljnoj arhitekturi za paralelno izvršavanje metaheurističkih metoda, bitni aspekti su komunikacija i sinhronizacija između paralelnih procesa (Blaise, 2012). Kada se koriste sistemi sa deljenom memorijom (slika 1) nema fzičkog transfera podataka između procesora. Svi procesori mogu da pristupe zajedničkoj (deljenoj) memoriji i da u nju upisuju ili iz nje čitaju podatke. U tom slučaju, sinhronizacija koraka izvršavanja predstavlja glavnu teškoću. Naime, pošto može doći do simultanog pristupa zajedničkoj memoriji, važno je da se osigura da se relevantne informacije pravilno koriste. Preciznije, važno je da se obezbedi da se dati podatak ne može pročitati pre nego što se njegova vrednost upiše na odgovarajuću memorijsku lokaciju, kao i da ne bude prepisan pre nego što su ga svi zainteresovani procesori preuzeli. Barijere i semafori su najčešće kontrolne promenljive koje se koriste za sinhronizaciju procesa.



Slika 1: Višeprocesorski sistem sa deljenom memorijom

U slučaju višeprocesorskih sistema kod kojih procesori imaju privatne memorije, komunikacija se sastoji u razmeni informacija između različitih procesora. Svaki procesor u svojoj lokalnoj memoriji čuva kopije podataka relevantnih za procese koji se na njemu izvršavaju. Dakle, potrebno je da se izvrši fizički prenos podataka iz memorije jednog procesora (pošiljaoca) u memoriju drugog (primaoca) da bi primalac bio svestan promena nastalih u vrednostima odgovarajućih podataka. Komunikacioni kanali (linkovi) spadaju u ulazno/izlazne komponente koje su po pravilu najsporiji delovi računara. Zato uvek treba imati na umu da prenos podataka izaziva komunikaciona kašnjenja koja mogu značajno pogoršati performanse paralelnog izvršavanja. Minimizacija komunikacionih kašnjenja može se postići smanjivanjem količine i/ili učestalosti razmene podataka, kao i izborom najpodesnije topologije veza među procesorima (slika 2). potpuno povezana arhitektura (slika 2a) je najpoželjnija, ali je teška za implementaciju kad broj procesora raste. Najčešće korišćena topologija višeprocesorskih sistema (zvezdasta arhitektura) ilustrovana je na slici 2b. Ona je najlakša za realizaciju, a komunikaciono kašnjenje nije preveliko jer je rastojanje između bilo koja dva procesora najviše dva. Koristi se za implementaciju centralno koordinisanih paralelnih aplikacija ili asinhronih kooperativnih izvršavanja koja zahtevaju globalnu memoriju. Prstenasta arhitektura se sve češće koristi za distribuirano (necentralizovano) izvršavanje koje ne podrazumeva globalnu memoriju. Osnovne prednosti i mane raznih višeprocesorskih topologija analizirane su u Cvetković and Davidović 2011 korišćenjem spektralne teorije grafova.



Slika 2: Različite arhitekture višeprocesorskih sistema sa privatnim memorijama

Pregledom novije literature, može se zaključiti da su višeprocesorski sistemi sa privatnim memorijama rasprostranjeniji od sistema sa deljenom memorijom. Razlog tome je najverovatnije jednostavnost primene kao i veća skalabilnost. Multikompjuteri se javljaju i u savremenim višeprocesorskim sistemima koji sadrže hiljade procesora umreženih u GRID i CLOUD sisteme (Teijeiro et al. 2016). Nedavno su se pojavili paralelni sistemi zasnovani na velikom broju jednostavnih procesorskih elemenata koji omogućavaju izvršavanje ogromnog broja jednostavnih operacija. To su tzv. GPU procesori, prvenstveno namenjeni za obradu slika, ali su se pokazali korisnima i za druge aplikacije. U poslednje vreme postali su popularni i za razvoj paralelnih metaheuristika (Alba *et al*. 2013). Osnovna ideja kod GPU sistema je da se glavni algoritam izvršava na centralnoj procesorskoj jedinici, a složena izračunavanja se po SIMD modelu dele na GPU i izvršavaju paralelno.

3. MERE PERFORMANSI ZA PARALELNE METAHEURISTIKE

Metaheuristike najčešće predstavljaju stohastičke procese pretrage. To znači da se uzastopnim izvršavanjem odgovarajuće metode nad istim skupom podataka ne dobijaju uvek ista rešenja. Dodatno, paralelizacijom se menja struktura samog algoritma, pa se tako može dobiti potpuno nova metoda pretraživanja (Alba 2005). U

takvoj situaciji, strandardne mere performasi paralelnih algoritama (ubrzanje i efikasnost) više ne mogu da se koriste, pa je važno pravilno definisati nove mere za procenu kvaliteta paralelnih verzija metaheuristika. Najčešće korišćena mera je kombinacija kvaliteta rešenja i vremena potrebnog da se konačno rešenje pronađe (Alba 2005, Crainic and Toulouse 2010). Naime, kod većine paralelnih implementacija, rešenja koja se dobiju sekvencijalnom i paralelnom verzijom metaheurističke metode razlikuju se po kvalitetu ali i strukturi. Dakle, osnovni cilj paralelizacije je da se dobije metoda koja nadmašuje sekvencijalnu verziju kako u pogledu kvaliteta rešenja tako i u računskoj efikasnost. Preciznije, smatramo da je paralelna varijanta uspešnija ukoliko generiše rešenje istog kvaliteta za kraće vreme ili ukoliko u istom vremenu izvršavanja dobije kvalitetnije rešenje. To se formalno može opisati sledećom formulom

$$U_q = S\frac{f_a}{f_q} \tag{1}$$

pri čemu S predstavlja faktor standardnog ubrzanja algoritma dobijenog paralelizacijom, f_1 je vrednost funkcije cilja koja odgovara sekvencijalnom izvršavanju, a f_q vrednost funkcije cilja koja se dobija paralelnim izvršavanjem na q procesora.

Ovako definisana mera će skalirati faktor ubrzanja imajući na umu uštede koje se odnose ne kvalitet rešenja: ako je rešenje poboljšano, faktor ubrzanja množi se koeficijentom čija vrednost je veća od jedan (značenje toga je da je paralelno izvršavanje još efikasnije, tj. korisnije); sa druge strane, ako je kvalitet rešenja degradiran, smanjuje se efikasnost paralelnog izvršavanja (jer će se faktor ubrzanja množiti brojem koji je manji od jedinice). Ako se sekvencijalnim i paralelnim izvršavanjem dobija isto rešenje, mera efikasnosti paralelnog izvršavanja svodi se na faktor ubrzanja.

4. PARALELIZACIJA METAHEURISTIKA KOJE KORISTE JEDNO REŠENJE

Metaheuristike koje koriste jedno rešenje najčešće se sastoje iz dva koraka: neke verzije lokalnog pretraživanja (koje im omogućava da popravljaju početno rešenje) i perturbacije rešenja (kojom se izlazi iz zamke lokalnog optimuma). Tipični primeri ovih metaheuristika su simulirano kaljenje (SA), tabu pretraživanje (TS), metoda promenljivih okolina (VNS) i adaptivna procedura slučajne pohlepne pretrage (GRASP) (Gendreau and Potvin 2010).

Paralelizacija ovih metoda može se realizovati na raznim nivoima. Najjednostavnije je da se okoline podele na odgovarajući broj delova i da svaki procesor pretražuje u svom delu okoline. Takva paralelizacija spada u kategoriju 1C/*/SPSS, pri čemu se srednje polje popunjava sa RS ili C zavisno od toga da li se pretraživanje vrši sinhrono ili asinhrono. To je paralelizacija niskog nivoa sa fino granulisanim modulima koji se paralelno izvršavaju. Viši nivo granulacije postiže se kada se paralelizuju pretraživanja različitih okolina. Naime, svaki od paralelnih procesora može da izvrši perturbacioni korak i lokalnu pretragu iz dobijenog rešenja. Time se implementira strategija 1C/*/MPSS sa istim mogućnostima za vrednosti srednjeg polja kao i u prethodnom slučaju. U oba ova slučaja sličnosti sekvencijalne i paralelnih verzija metode su velike. Preciznije, izvršava se uvek ista varijanta metaheurističke metode samo su neki koraci izmešani. Najuopštenija paralelizacija visokog nivoa je kada se paralelno izvršavaju različite varijante sekvencijalne metaheurističke metode. One se najčešće razlikuju po vrednosti parametara, a svakako po vrednosti seed-a, parametra koji se koristi za inicijalizaciju generatora slučajnih brojeva. Izabrane varijante metode mogu se izvršavati nezavisno, što simulira multistart varijantu i samo skraćuje vreme potrebno za izvršavanje svih restarta. Mnogo kompleksnije, ali i korisnije je kada postoji interakcija (kooperacija) između tih varijanti, tj. razmena relevantnih informacija kojom se pretraga usmerava na okoline kvalitetnijih rešenja. U najširem smislu, ova paralelizacija klasifikuje se kao pC/C/MPDS.

Tabela 1: Strategije za paralelizaciju VNS metode

Strategija	Klasifikacija	Detalji
IVNS	pC/RS/MPDS	nezavisna izvršavanja različitih VNS metoda, centralizovano
PVNSPLS	1C/C/SPSS	fino granulisano: par. LS u sekv. VNS
DVNS	1C/C/MPSS	srednja granulacija: paralelno SH+LS za različito *k* isto LS
CVNS1	pC/C/SPDS	srednja granulacija: paralelno SH+LS za različito *k* i različito LS
CVNS2	pC/C/SPDS	krupna granulacija: iteracije se izvršavaju paralelno
CVNSring	pC/C/MPDS	paralelno SH+LS za različito *k* i različito LS, ne-centralizovano

VNS metoda predložena je prvi put u radu Mladenović and Hansen 1997. Uspešno je primenjena na mnoge optimizacione probleme (Hansen *et al*. 2010). Paralelizacija VNS metode razmatrana je u mnogim radovima, neke implementacije sumirane su u Crainic *et al*. 2014. U radovima Davidović and Crainic 2012, Davidović and Crainic 2013, Davidović and Crainic 2015 predložene su strategije sumirane u tabeli 1.

5. PARALELIZACIJA METAHEURISTIKA ZASNOVANIH NA POPULACIJI REŠENJA

Metaheuristike koje rade sa populacijom rešenja su stohastičke tehnike pretraživanja koje su uspešno primenjene na razne složene probleme optimizacije. Ove metode su iterativne tehnike koje koriste stohastičke operatore varijacije (transformacije) nad skupom jedinki (populacijom). Svaka jedinka u populaciji je kodirana verzija dopustivog rešenja. Funkcija evaluacije definiše vrednost prilagođenosti (fitnesa) kojom se za svaku jedinku određuje kvalitet odgovarajućeg rešenja. Poznate metaheurističke metode iz ove grupe su evolucioni algoritmi (EA), optimizacija kolonijom mrava (ACO), optimizacija rojem čestica (PSO), diferencijalna evolucija (DE), kolonija veštačkih pčela (ABC) i optimizacija kolonijom pčela (BCO)) (Gendreau and Potvin 2010).

Ova vrsta metaheuristika sadrži visok stepen inherentnog paralelizma jer su izračunavanja vezana za svaku jedinku u populaciji nezavisna (Alba *et al*. 2013). Time se ostvaruje paralelizacija niskog nivoa (fine granulacije) pri čemu se paralelno računaju, na primer, operatori varijacije i vrednosti za funkciju prilagođenosti svake jedinke. Opisana paralelizacija spada u kategoriju 1C/RS/MPSS. Drugi nivo paralelizacije, iz kategorije pC/RS/MPSS, postiže se deljenjem populacije na nekoliko pod-populacija koje se na različitim procesorima tretiraju nezavisno do nekog određenog momenta u kome dolazi do komunikacije među procesorima i razmene (migracije) jedinki. Proces obrade pod-populacija i komunikacija smenjuje se do zadovoljenja nekog kriterijuma zaustavljanja. Ovaj način paralelizacije u literaturi je poznat kao model ostrva (islands model). Naravno, paralelizacija visokog nivoa pC/C/MPDS (kada se istovremeno izvršavaju različite sekvencijalne varijante date metaheurističke metode) i ovde je moguća.

BCO metoda predložena je prvi put u radu Lučić and Teodorović 2001. Uspešno je primenjena na mnoge optimizacione probleme (Davidović 2015, Teodorović *et al*. 2015). Paralelizacija BCO metode razmatrana je u radovima Davidović *et al.* 2011 i Davidović *et al.* 2013, a predložene strategije sumirane su u tabeli 2.

Tabela 2: Strategije za paralelizaciju BCO metode

Name	Description	Classification	Details
DBCO	same BCO, different seed	pC/RS/MPSS	reduced stopping criteria
BBCO	same BCO, different seed	pC/RS/MPSS	reduced number of bees
MBCO	different BCO, independent	pC/RS/MPDS	different B and NC
CBCO1	cooperative synchronous	pC/KS/MPDS	fixed number of communications
CBCO2	cooperative synchronous	pC/KS/MPDS	variable number of communications
GBCO1	cooperative asynchronous	pC/C/MPDS	centralized communications
GBCO2	cooperative asynchronous	pC/C/MPDS	non-centralized communications

6. ZAKLJUČAK

U radu su opisane specifičnosti vezane za paralelizaciju metahurističkih metoda. Ilustrovane su neke od najpopularnijih višeprocesorskih arhitektura koje se koriste za paralelno izvršavanje metaheuristika i sumirane su paralelne implementacije VNS i BCO metoda koje su rezultat istraživanja autora. U skorije vreme su se pojavili i računari zasnovani na protoku podataka (Data Flow), tzv. FPGA (Field Programmable Gate Arrays), kod kojih je algoritam ukodiran u hardver tako da se istovremeno izvršavaju različite instrukcije nad različitim podacima. Paralelizacija metaheuristika za ovakve sisteme još uvek je veliki izazov za istraživače.

LITERATURA

[1] Alba E. (Ed.) (2005). *Parallel metaheuristics: a new class of algorithms*. John Wiley & Sons.

[2] Alba, E., Luque, G., & Nesmachnow, S., (2013). Parallel metaheuristics: recent advances and new trends. *International Transactions in Operational Research*, *20*(1), pp.1-48.

[3] Blaise, B., (2012). *Introduction to Parallel Computing*, https://computing.llnl.gov/tutorials/parallel_comp/.

[4] Crainic, T. G., & Hail, N. (2005). Parallel meta-heuristics applications. In E. Alba (Ed.), *Parallel Metaheuristics: A new Class of Algorithms* (pp. 447-494). Hoboken, NJ: John Wiley & Sons.

[5] Crainic, T. G., & Toulouse, M. (2010). Parallel Meta-heuristics. In M. Gendreau, J.Y. Potvin (Eds.), *Handbook of metaheuristics* (pp. 497-541). USA: Springer Science+Business Media.

[6] Crainic, T.G., Davidovic, T., & Ramljak, D., 2014. Designing parallel meta-heuristic methods. *High Performance and Cloud Computing in Science and Education, IGI-Global*, pp.260-280.

[7] Cvetković, D., & Davidović, T. (2011). Multiprocessor Interconnection Networks. In D. Cvetković, I. Gutman (Eds), *Selected Topics on Applications of Graph Spectra, second edition, Vol: 14(22)*, (pp. 35-62). Serbia: Mathematical Institute SANU.

[8] Davidović, T., & Crainic, T. G. (2012). MPI parallelization of variable neighborhood search. *Electronic Notes in Discrete Mathematics*, *39*, 241-248.

[9] Davidović, T., & Crainic, T. G. (2013a). Parallelization strategies for variable neighborhood search, CIRRELT-2013-47, 1-32. https://www.cirrelt.ca/DocumentsTravail/CIRRELT-2013-47.pdf

[10] Davidović, T., & Crainic, T.G., 2015. Parallel local search to schedule communicating tasks on identical processors. *Parallel Computing*, *48*, pp.1-14.

[11] Davidović, T., Jakšić, T., Ramljak, D., Šelmić, M., & Teodorović, D. (2013). Parallelization strategies for bee colony optimization based on message passing communication protocol. *Optimization: A Journal of Mathematical Programming and Operations Research, 62(8)*, 1113-1142.

[12] Davidović, T., Ramljak, D., Šelmić, M., & Teodorović, D. (2011). MPI parallelization of bee colony optimization, In *1st International Symposium & 10th Balkan Conference on Operational Research, Vol: 2*, (pp. 193-200). Thessaloniki: University of Macedonia, Economic and Social Sciences.

[13] Davidović, T., Teodorović, D. , & Šelmić, M. (2015). Bee colony optimization Part I: The algorithm overview. *Yugoslav Journal of Operations Research*, *25*(1) 33-56.

[14] Gendreau, M., & Potvin, J. Y. (2010). *Handbook of metaheuristics*, (second edition) Springer.

[15] Hansen, P., Mladenović, N., Brimberg, J., & Moreno-Pérez, J. A. (2010). Variable neighbourhood search. In M. Gendreau, J-Y. Potvin (Eds.), *Handbook of Metaheuristics,* (second edition) (pp. 61-86). USA: Springer Science+Business Media.

[16] Lučić, P., & Teodorović, D. (2001). Bee system: modeling combinatorial optimization transportation engineering problems by swarm intelligence. *In Preprints of the TRISTAN IV Triennial Symposium on Transportation Analysis* (pp. 441-445). Sao Miguel, Azores Islands.

[17] Mladenović, N., & Hansen, P. (1997). Variable neighborhood search. Computers & Operations Research, *24(11)*, 1097-1100.

[18] Smutnicki, C., & Bożejko, W., (2015), February. Parallel and Distributed Metaheuristics. In *International Conference on Computer Aided Systems Theory* (pp. 72-79). Springer International Publishing.

[19] Talbi, E.-G. (Ed.) (2009). *Metaheuristics: From Design to Implementation*. Hoboken, New Jersey: John Wiley & Sons, Inc.

[20] Teijeiro, D., Pardo, X.C., González, P., Banga, J.R., & Doallo, R., 2016. Towards cloud-based parallel metaheuristics: A case study in computational biology with Differential Evolution and Spark. *International Journal of High Performance Computing Applications*, p.1094342016679011.

[21] Teodorović, D., Šelmić, M., & Davidović, T. (2015). Bee Colony Optimization-part II: The application survey. *Yugoslav Journal of Operations Research*, *25*(2), 185-219.

[22] Verhoeven, M. G. A., & Aarts, E. H. L. (1995). Parallel local search. *Journal of Heuristics*, *1(1)*, 43-65.