Jedno rešenje distribuirane obrade zasnovane na proceni performanse računara

Igor Cavrić

***Sadržaj* — Danas, kada procesori stižu do granice uvećanja svog radnog takta, za izračunavanje složenih matematičkih operacija se obično koristi distribuirano računarstvo ili simetrični višeprocesorki računari. Primer programske podrške za višeprocesorke računare je Calculation Engine (CE) koji je razvijen od strane TelventDMS LLC iz Novog Sada. Na osnovama tog rešenja je izrađen ovaj rad kao mali doprinos distribuiranom računarstvu. Cilj rada je da se dobije kraće vreme obrade na klasteru komercijalno isplativih računara od skupog NUMA računara. U radu je opisano rešenje distribuirane obrade blokova elektro-distributivne mreže na klasteru poznavajući performanse računara.**

***Ključne reči* — balansiranje opterećenja, distribuirana obrada, procena performansi računara**

I. UVOD

IZRADA distribuiranih sistema velike skale integracije (eng. large-scale), poput elektro-distributivne mreže većeg grada ili regije, smatra se veoma važnim zadatkom u inženjerstvu sistema zasnovanih na računarima. Bitne osobina takvog distribuiranog sistema upravljanja navedene u [1]-[2] su: veliki broj ulaznih promenljivih (na pr. 13 miliona u državi Teksas, SAD), kratak vremenski period očitavanja ulaznih vrednosti i upravljanja sistemom u realnom vremenu (obično oko nekoliko sekundi), kratko vreme ispravke u slučaju greške (oko 2-3 sekunde), upravljanje zasnovano na složenim matematičkim operacijama (tradicionalno napisanih kao sekvencijalni program, obično na Fortranu). Uzevši u obzir činjenicu da današnji procesori stižu do granice uvećanja svog radnog takta, počelo se sa istraživanjem novih tehnoloških mogućnosti. Moguća rešenja su procesori sa više jezgara ili raspodela zadataka na više računara u mreži gde se može postići veliko ubrzanje obrade složenih matematičkih proračuna. Primer prvog rešenja je podsistem Calculation Engine (CE) koji je razvijen za rad na NUMA arhitekturi [3]. Za razliku od podsistema CE koji radi na računaru sa NUMA arhitekturom rešenje ovog rada za cilj ima distribuiranu obradu na klasteru ličnih računara (eng. personal computer) povezanih Eternet mrežom. Zajedničko za oba rešenja je poziv fortranske funkcije za obradu tokova snaga (eng. load flow) blokova elektro-distributivne mreže. Pre poziva, potrebno je na određen način urediti argumente funkcije. Svi podaci koji određuju jedan blok se uređuju posebnim modulima za pripremu podataka; dodatno se dodaju podešavanja funkcije. Podaci koji čine jedan blok elektro-distributivne mreže se dele na podatke unutrašnjeg modela (eng. internal model) i modela stanja (eng. state model). Podaci unutrašnjeg modela su statički podaci i uređeni su kao niz bajtova koji predstavlja izgled bloka elektro-distributivne mreže. Stanje modela su dinamički podaci i to su vrednosti očitane sa SCADA-e i prethodni rezultat proračuna tokova snaga. Sledi opis arhitekture rešenja i komponente raspoređivača sa strategijama balansiranja opterećenja. Nakon toga je opisana komponenta izvršilac sa testom performanse (eng. benchmark). Konačno, su predstavljeni rezultati obrade Progress Energy Carolinas (PEC) sheme elektro-distributivne mreže i opšti zaključak o radu.

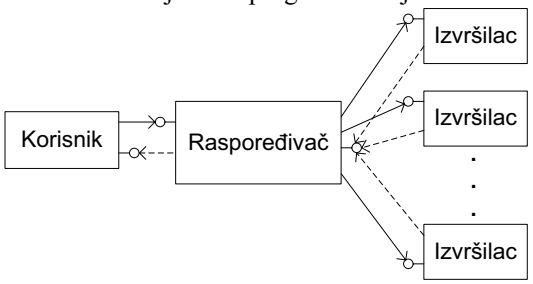
Ovaj rad je delimično finansiran od Ministarstva za nauku Republike Srbije, projekat 12004, od 2008. god.

Igor Cavrić, Fakultet tehničkih nauka u Novom Sadu, Fruškogorska 11, 21000 Novi Sad, Srbija (telefon: 381-21-4801100, e-mail: IgorCavric@gmail.com)

Mentori:
Miroslav Popović, Fakultet tehničkih nauka u Novom Sadu, Fruškogorska 11, 21000 Novi Sad, Srbija (telefon 381-21-4801100, e-mail: Miroslav.Popovic@rt-rk.com)
Ilija Bašičević, Fakultet tehničkih nauka u Novom Sadu, Fruškogorska 11, 21000 Novi Sad, Srbija (telefon 381-21-4801100, e-mail: basicevic@krt.neobee.net)
Branislav Atlagić, Fakultet tehničkih nauka u Novom Sadu, Fruškogorska 11, 21000 Novi Sad, Srbija (telefon 381-21-4801100, e-mail: Branislav.Atlagic@rt-rk.com)

II. OPIS PROGRAMSKE PODRŠKE

Sistem sa Sl. 1 se sastoji iz jedne komponente raspoređivača i više komponenti izvršioca. Komunikacija između komponenti je omogućena upotrebom *Windows Communication Foundation* aplikativne programske sprege sa *netTCP* povezivanjem [4]. Sve programske komponente su razvijene na programskom jeziku C#.



Sl. 1. Arhitektura rešenja.

A. Raspoređivač

Raspoređivač je povezan sa svakim izvršiocem, dok izvršioci nisu međusobno povezani. Raspoređivač je glavni čvor sistema čija je uloga da prima podatke i zadatke od korisnika koje raspoređuje izvršiocima. Za prijem podataka služe dva modula dok za prijem zadataka služi jedan modul. Svi moduli komuniciraju sa korisničkom

stranom na različitim prolazima (eng. port). Svaki od modula ima svoje delegate preko kojih obaveštava ostale module o pristiglim podacima ili zadacima. Raspoređivač se sastoji iz dva raspoređivača: raspoređivača podataka i raspoređivača zadataka. Raspoređivač podataka statičke podatke prosleđuje svim izvršiocima. Dinamički podaci se mogu slati modulu za prijem dinamičkih podataka ili upakovni u zadatak modulu za prijem zadataka. Raspoređivač zadataka u zavisnosti od strategije odlučuje o izboru jednog izvršioca kojem se dodeljuje zadatak. Podaci o svim izvršiocima se čuvaju u posebnom modulu. Pored raspoređivanja, komponenta raspoređivača stvara i nadzire procese izvršioca na drugim računarima.

Raspoređivač dodeljuje zadatak jednom izvršiocu u zavisnosti od izabrane strategije balansiranja opterćenja. Balansiranje opterećenja (eng. Load Balansing) je tehnika ravnomerne raspodele zadataka između dva ili više računara, mrežnih čvorova, procesora ili drugih resursa čiji je cilj dobijanje optimalne opterećenosti resursa, povećanje protoka, smanjenje odziva ili izbegavanje preopterećenja. Usluga balansiranja opterećenja zavisi od strategije balansiranja opterećenja.

Strategija kružne dodele zadataka čuva jedinstvene oznake svih pokrenutih izvršilaca u jednoj listi. Zadaci se dodeljuju redom izvršiocima iz liste počev od prvog izvršioca. Zadatak se dodeljuje izvršiocu samo ako nije zauzet. Ukoliko je izvšilac zauzet zadatak se dodeljuje sledećem slobodnom izvršiocu. U slučaju da ni jedan izvršilac nije slobodan kroz listu se kruži dok se ne pojavi prvi slobodan izvršilac.

Strategija favorizacije najboljeg izvršioca čuva jedinstvene oznake svih pokrenutih izvršilaca uređene prema indeksu performanse izvršioca od najboljeg ka najlošijem. Zadaci se dodeljuju prvom (najboljem) izvršiocu iz liste sve dok je u stanju slobodan. Ukoliko je izvršilac u stanju zauzet zadatak se dodeljuje sledećem izvršiocu iz liste. Pri raspoređivanju svakog zadatka pretraga slobodnog izvršioca počinje sa početka liste. Razlika između strategije kružne dodele i strategije favorizacije najboljeg izvršioca je što prva strategija traži slobodnog izvršioca od sledećeg elementa liste dok druga uvek kreće s početka liste.

B. Korisnik

Korisnik izdaje zadatke ali ima i ulogu uslužioca (eng. server) podataka. Na početku rada sistema korisnik do raspoređivača šalje sve blokove statičkih podataka koje raspoređivač prosleđuje svim izvršiocima. Blokovi dinamičkih podataka (koji su po veličini znatno manji od blokova statičkih podataka) se šalju raspoređivaču zajedno sa zadatkom i podešavanjima funkcije.

C. Izvršilac

Izvršilac prihvata blokove statičkih podataka i zadatke (u koje su upakovani i blokovi dinamičkih podataka). Blokovi statičkih podataka se čuvaju u modulu za statičke podatke. Po pristizanju zadatka izvršilac uređuje blokove statičkih i blokove dinamičkih podataka sa podešavanjima funkcije čime uređuje argumente funkcije za obradu. Priprema i obrada zadatka se obavljaju u dve programske niti (eng. thread). Zadatak se ne može obraditi dok nije pripremljen.

Izvršilac periodično očitava vrednost iskorištenosti procesorskog vremena (eng. processor time) i broja zadataka u procesorskom redu zadataka (eng. processor queue length) i računa indeks opterećenja. U cilju boljeg kvaliteta indeksa opterećenja D. Ferari u [5] je uveo nekoliko poželjnih osobina indeksa opterećenja. Dobar indeks opterećenja treba da:

- Predstavlja kvalitetnu procenu trenutnog opterećenja nekog čvora;
- Predvidi opterećenje u skorijoj budućnosti s obzirom da će na obradu zadatka više uticati buduće opterećenje nego trenutno;
- Bude relativno otporan na česte promene opterećenja.

Nakon niza ogleda Ferari je pokušao da zadovolji sve navedene osobine i kao rezultat tog napora dobio sledeću formulu:

$$Q_i = Q_{i-1}(1 - e^{-T}) + q_i e^{-T}, i \geq 1, Q_0 = 0 \qquad (1)$$

gde je Q vrednost indeksa, T interval usrednjavanja u sekundama, q vrednost iskorištenosti resursa.

Koristeću formulu (1) izvršilac dobija vrednosti indeksa opterećanja procesorskog vremena i procesorskog reda zadataka. Granične vrednosti oba indeksa opterećenja postavlja korisnik. Ukoliko oba indeksa opterećenja pređu zadatu granicu izvršilac je u stanju zauzet dok je u suprotnom u stanju slobodan.

Pored stanja, izvršilac je određen i indeksom performanse računara na kojem je pokrenut. Po stvaranju procesa izvršioca pokreće se niz provera po uzoru na *Whetstone* [6] test performansi (eng. benchmark). Proverava se vreme obrade sledećih operacija:

- Izračunavanja jednačina, gde su početne vrednosti $x_1 = x_2 = x_3 = x_4 = 1$:

$$\begin{aligned} x_1 &= \frac{1}{2}(x_1 + x_2 + x_3 - x_4) \\ x_2 &= \frac{1}{2}(x_1 + x_2 - x_3 + x_4) \\ x_3 &= \frac{1}{2}(x_1 - x_2 + x_3 + x_4) \\ x_4 &= \frac{1}{2}(-x_1 + x_2 + x_3 + x_4) \end{aligned} \qquad (2)$$

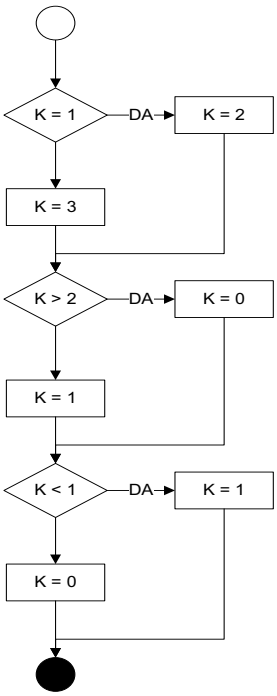
- Izračunavanja trigonometrijske funkcije, gde su početne vrednosti $x = y = 0{,}5$ i $t = 0{,}499975$:

$$\begin{aligned} x &= t \cdot \arctan\left(\frac{2 \cdot \sin x \cdot \cos x}{\cos(x \cdot y) + \cos(x - y) - 1}\right) \\ y &= t \cdot \arctan\left(\frac{2 \cdot \sin y \cdot \cos y}{\cos(x \cdot y) + \cos(x - y) - 1}\right) \end{aligned} \qquad (3)$$

- Uobičajenih matematičkih funkcija, gde su početne vrednosti $x = 0{,}5$ i $t = 0{,}50025$:

$$x = \sqrt{e^{\frac{\ln x}{t}}} \quad (4)$$

- Stvaranja i uređivanja (eng. sort) niza nasumično odabranih vrednosti.
- Uslovnih programskih skokova Sl. 2, gde je početna vrednost $K = 1$:



Sl. 2. Blok dijagram algoritma uslovnih skokova.

III. Rezultati

Merenja su obavljena u laboratoriji sa heterogenom strukturom računara povezanih Eternet mrežom protoka 100 Mbit/sec. Ukupno se koristi deset računara od kojih su osam izvršioci i po jedan raspoređivač i korisnik. Struktura računara sa ulogom koja im je dodeljena i indeksom performanse je prikazana u Tabeli 5.1. Operativni sistem na svim računarima je *Microsoft Windows XP Professional* sa *ServicePack2*.

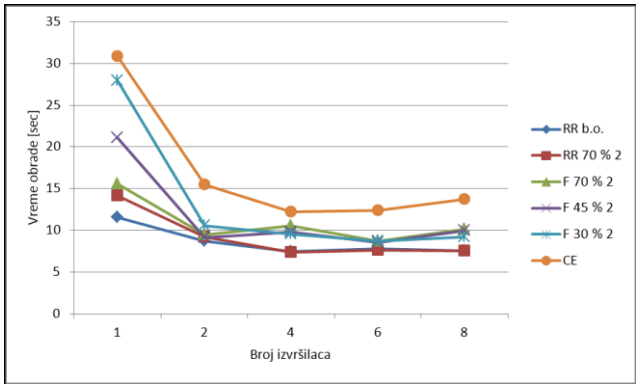
Tabela 1: Struktura računara u laboratoriji.

Redni broj	***Procesor***	***Indeks perf.***	***Uloga***
1	Intel Celeron 1,7 GHz	525	Izvršilac 1
2	Intel Pentium 4 2 GHz	440	Izvršilac 2
3	Intel Pentium 4 2 GHz	445	Izvršilac 3
4	Intel Pentium 4 2,4 GHz	366	Izvršilac 4
5	Intel Pentium 4 2 GHz	-	Korisnik
6	Intel Pentium 4 2 GHz	-	Raspoređ.
7	Intel Pentium 4 1,7 GHz	525	Izvršilac 5
8	Intel Pentium 4 2,4 GHz	352	Izvršilac 6
9	Intel Celeron 1,7 GHz	529	Izvršilac 7
10	Intel Pentium 4 2,6 GHz	326	Izvršilac 8

Merenja su obavljena za sistem sa jednim, dva, četiri, šest i osam izvršilaca. Takođe, mereno je i rešenje *Calcluation Engine*. Rezultat merenja je ukupno vreme obrade 55 zadataka PEC sheme elektro-distributivne mreže. Za svaki broj izvršilaca obavljena su merenja za obe strategije balansiranja opterećenja i više graničnih vrednosti opterećenosti izvršioca:

- Strategijom kružne dodele bez ograničenja;
- Strategijom kružne dodele kada je granična vrednost iskorištenosti procesorskog vremena (eng. Processor Time) 70 % i broj zadataka u procesorskom redu zadataka (eng. Processor Queue Length) 2;
- Strategijom favorizacije najboljeg izvršioca kada je granična vrednost iskorištenosti procesorskog vremena 70 % i broj zadataka u procesorskom redu zadataka 2;
- Strategijom favorizacije najboljeg izvršioca kada je granična vrednost iskorištenosti procesorskog vremena 45 % i broj zadataka u procesorskom redu zadataka 2;
- Strategijom favorizacije najboljeg izvršioca kada je granična vrednost iskorištenosti procesorskog vremena 30 % i broj zadataka u procesorskom redu zadataka 2.

Rezultati merenja (Tabela 2 i Sl.3) pokazuju da CE ima najveće vreme obrade zadataka zbog nedeljenja podataka na statičke i dinamičke. Međutim, povećanjem broja izvršilaca to vreme se smanjuje i do dva i po puta. S druge strane, najkraće vreme obrade se dobija upotrebom strategije kružne dodele. Strategija favorizacije najboljeg izvršioca i više zavisi od strukture izvršioca i graničnih vrednosti nego od broja izvršilaca. Kod upotrebe jednog izvršioca vidi se da se vreme obrade povećava sa pooštravanjem graničnog uslova, jer se na raspodelu nekih zadataka čeka dok jedini izvršilac ne pređe u stanje slobodan. Povećanjem broja izvršioca raste i mogućnost raspodele tih zadataka drugim izvršiocima zbog čega se vreme obrade smanjuje. Međutim, primeti se da je vreme obrade upotrebom šest izvršilaca kraće od vremena kada je u upotrebi osam izvršilaca iako osmi izvršilac ima najbolji indeks performanse. Razlog tome je što test performanse proverava procesorsku moć računara, ali ne i mrežnu opremu kao ni mrežne vodove do tog računara koje mogu dovesti do smanjenja brzine prenosa a tim i ukupnog vremena obrade zadataka.



Sl. 3. Uporedni prikaz vremena obrade svih merenja.

TABELA 2: UPOREDNI PRIKAZ VREMENA OBRADE SVIH MERENJA.

Br. Izvr	*RR b.o.*	*RR 70%2*	*F 70%2*	*F 45%2*	*F 30%2*	*CE*
1	11.55	14.1	15.58	21.09	27.89	30.89
2	8.73	9.22	9.43	9.15	10.52	15.5
4	7.44	7.38	10.55	9.77	9.53	12.24
6	7.76	7.61	8.67	8.56	8.71	12.37
8	7.56	7.5	10.09	9.99	9.18	13.68

U toku merenja su periodično beleženi parametri opterećenja izvršioca: iskorištenost procesorskog vremena i broj zadataka u procesorskom redu zadataka. Perioda beleženja je jedna sekunda. Pri upotrebi strategije kružne dodele vrednost oba parametra je uglavnom ujednačena, ali postoje i neka odstupanja poput vrednosti broja zadataka sedmog izvršioca za strategiju bez ograničenja. To je pojava koja se dešava usled operativnog sistema opšte namene i procesa nad kojima komponenta izvršioca nema nadzor. Kod strategije favorizacije najboljeg izvršioca očekivano je da su izraženije vrednosti parametara boljih izvršioca zbog većeg broja zadataka koje obrađuju.

IV. ZAKLJUČAK

U poređenju dve strategije sa nasleđenim rešenjem najbolje vreme obrade zadataka postignuto je upotrebom strategije kružne dodele zadataka. Međutim, pokazano je da veoma brzo, već na četiri izvršioca, dolazi do zasićenja ubrzanja. Razlog tome je manje vreme obrade od vremena prenosa podataka čime sa povećanjem broja izvršioca dolazi do čekanja izvršioca na zadatak. Strategija favorizacije najboljeg izvršioca je postigla najbolje vreme obrade od 8,67 sekundi što je sekundu sporije od najboljeg vremena strategije kružne dodele zadataka.

Dalji pravci razvoja rešenja su usmereni ka usavršavanju ispitivanja performanse u koji bi bila uključena i provera kvaliteta mrežne opreme. Osim toga, izmenom strategije favorizacije najboljeg izvršioca tako da naizmenično favorizuje više izvršilaca bi se dobilo kraće vreme obrade za tu strategiju.

LITERATURA

[1] M. Popovic, I. Basicevic, and V. Vrtunski, "A Task Tree Executor: New Runtime for Parallelized Legacy Software", Engineering of Computer Based System Conference (ECBS), San Francisco, USA, April 2009

[2] I. Basicevic, S. Jovanovic, B. Drapsin, M. Popovic, and V. Vrtunski, "An Approach to Parallelization of Legacy Software", IEEE ECBS-EERC, Novi Sad, Serbia, Sept. 2009

[3] B. Trivunovic, M. Popovic, V. Vrtunski, "An Application Level Parallelization of Complex Real-Time Software", 17th IEEE International Conference and Workshops on Engineering of Computer Based Systems (ECBS), UK, 2010

[4] S. Klein, "Professional WCF programming: .NET development with the Windows communication foundation", Wiley Publishing, Indianapolis, 2007

[5] D. Ferrari, S. Zhou, "An Empirical Investigation of Load Indices for Load Balancing Applications", Berkeley, 1988

[6] H.J. Curnow, B.A. Wichmann, "A Synthetic Benchmark" in Computer Journal, Vol. 19, No 1, 1976, pp. 43-49.

ABSTRACT

Under the current circumstances, when the processors are facing the well-known frequency wall, modern symmetric multiprocessors or distributed computing are used to calculate complex mathematical operations. An example of the software architecture for a multiprocessor computer is the Calculation Engine (CE) developed by TelventDMS LLC, Novi Sad. Based on the CE, this paper contributes to the overall research effort in the area of distributed computing. The aim of this paper is to have better results on cluster of commercially viable computers over an expensive NUMA computer. The results of these solutions that calculate load flow function of Progress Energy Carolinas (PEC) scheme are compared. Since the performance of distributed solution is better than CE, it is showed that a very large-scale distributed system can be built in the distributed computing technology.

A SOLUTION OF A TASK DISTRIBUTION BASED ON BENCHMARK

Igor Cavrić