

Poređenje CDCL i stohastičkih SAT rešavača

Automatsko rezonovanje – 2017/2018. školske godine

Marija Mijailović 1093/2017

Miroslav Mišljenović 407/2014

1 Uvod

Jedan od glavnih problema u računarstvu trenutno predstavlja problem zadovoljivosti. Taj problem se svodi na pronalaženje interpretacije koja zadovoljava određenu bulovsku formulu. Drugim rečima, potrebno je pronaći vrednosti promenljivih na takav način da celokupna formula bude tačna. Ovaj problem se još naziva i SAT problem (eng. satisfiability). SAT je prvi poznati primer NP-kompletnog problema. To znači da ne postoji poznati algoritam koji rešava sve instance SAT problema. Međutim, razvijeni su posebni SAT rešavači koji mogu rešiti dovoljno veliki skup problema. Oni su našli praktičnu primenu u raznim oblastima, kao na primer verifikaciji hardvera i softvera, automatskog dokazivanja teorema, ali i u svakodnevnim problemima poput pravljenja rasporeda časova ili rešavanja logičkih problema. Jedan od takvih logičkih problema je i problem rešavanja sudokua.

Postoje dve vrste SAT rešavača, CDCL (eng. Conflict-Driven Clause Learning) i stohastički rešavači. CDCL rešavači su zasnovani na unapređenoj verziji DPLL algoritma, dok stohastički imaju veliki broj različitih algoritama. Osnovna razlika je i to što CDCL rešavač, osim što može da pokaže da je formula zadovoljiva, može da javi da je formula nezadovoljiva, dok stohastički može da prijavi samo zadovoljivost formule. U ovom radu ćemo uporediti dva SAT rešavača iz ove dve grupe, minisat i ubcsat.

2 Sudoku

Sudoku je logička igra u kojoj se barata brojevima, a u nekim posebnim vrstama i slovima. Sastoji se od kvadratne ili pravougaone tabele sa poljima u koja se unose brojevi. Standardan sudoku se igra na tabeli od 9x9 polja, a cela tabela se sastoji od devet manjih kvadrata dimenzije 3x3. Analogno ovome, mogu se kreirati i veće tabele za igru, npr. 16x16, 25x25 i

36x36, ali su ovakvi problemi jako teški ako treba da ih rešava čovek. Brojevi su uvek od 1 do dimenzije sudokua. Na igraču je da popuni prazna polja pridržavajući se slededih pravila:

1. U svakom redu moraju biti različiti brojevi
2. U svakoj koloni moraju biti različiti brojevi
3. U svakom unutrašnjem kvadratu/pravougaoniku moraju biti različiti brojevi

Instanca je rešiva ako je moguće popuniti tabelu pridržavajući se ovih pravila.

Prva stvar koja je bitna prilikom generisanja instanci je da instanca mora imati jedinstveno rešenje. U skladu s tim, prvobitno zamišljeni princip nasumičnog popunjavanja tabele ne garantuje da će uslov rešivosti biti ispunjen, pa nam se kao rešenje nametnuo drugačiji pristup. Drugi pristup je bio generisanje popunjene tabele po poznatim uslovima, a zatim uklanjanje određenog broja brojeva na slučajan način dok se ne dođe do zadovoljavajuće popunjenosti. Kada se kreira sudoku koji će rešavati čovek, bitno je formirati ga tako da uvek ima jedinstveno rešenje. Zbog toga se koriste posebni algoritmi koji uklanjaju jedan po jedan broj i testiraju sudoku na jedinstvenost. Ukoliko u nekom trenutku sudoku postane nejedinstven, primenjuje se backtreking algoritam i pokušava se sa brisanjem drugog broja. Ipak, ovakva operacija može biti veoma zahtevna za formiranje sudokua čija je veličina veća od standardne.

Uobičajeno postoje tri težine sudokua, i to lak, srednje težak i težak. U literaturi se ne precizira kako se određuje težina sudokua. Jedna mera je da najteži sudoku ima barem 17 popunjenih polja, dok jednostavni sudokui imaju 30-35 popunjenih polja (misli se na 9x9 sudoku). Srednje težak je negde između ovakva dva.

Formirali smo više različitih veličina sudokua od standardnog 9x9, do velikog i računarski zahtevnog 25x25. Testiranje smo izvršili na dimenzijama 9x9, 16x16 i 25x25 i to po 3-4 različite instance da bi dobijeni rezultati bili pouzdani. Odlučili smo se za ove dimenzije, jer se sudoku tabele manjih dimenzija vrlo brzo rešavaju.

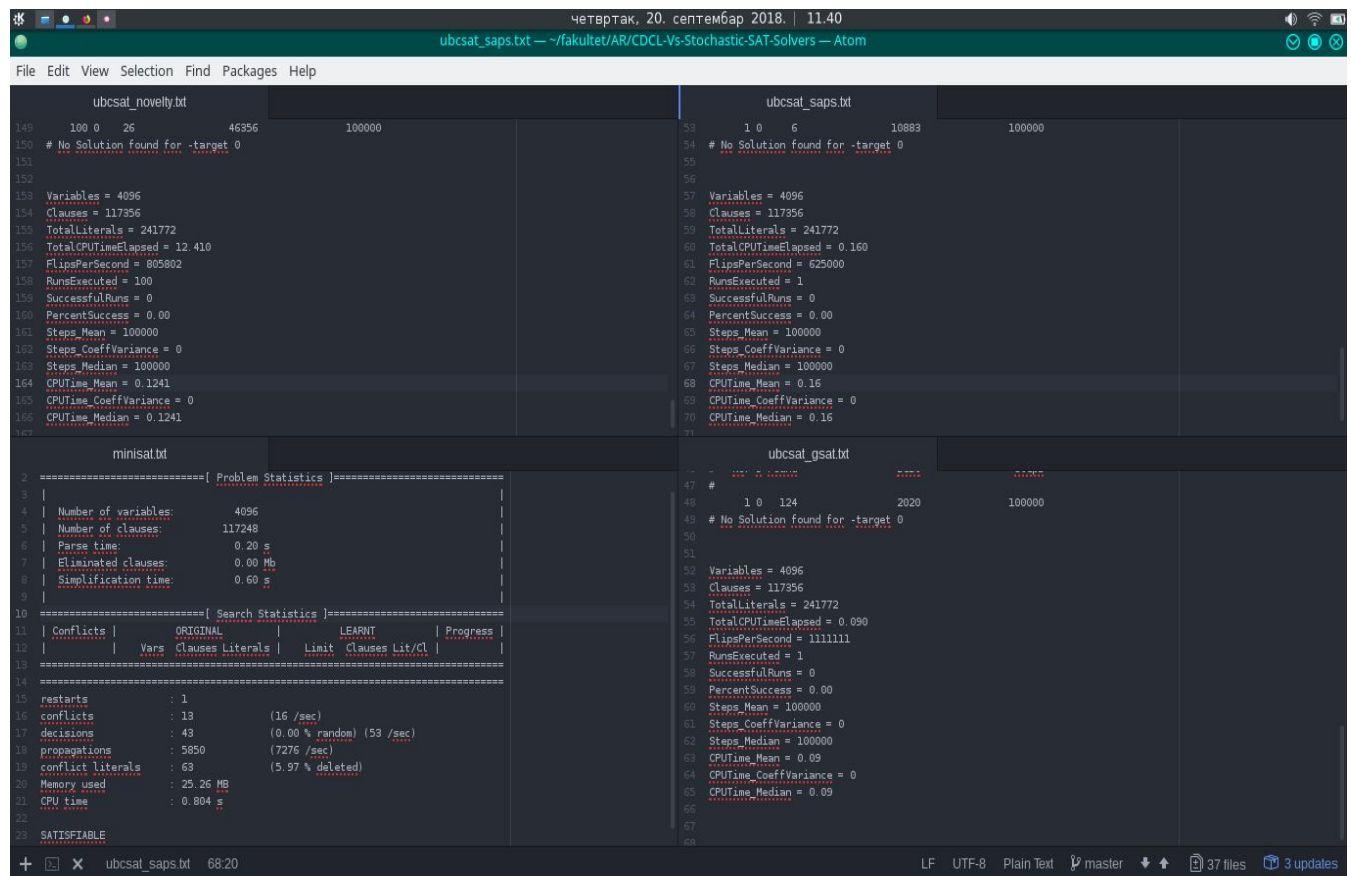
	9	6		4			3		1		8			6	4								8		6
	5	7	8	2							6		9		8		7	4		5	6	9			1
1			9				5		5											9			2	4	
		9		1					2	6	9	5				8		5					3		8
5												4		9						7	8		9	6	
4				9		6				8				2	7	9	1		9		2				3
		4			3			1								5				4	7			1	
				7	9	2	6		6		4		7		2				6			4	1	7	8
	2			5		9	8				1	2			9	3		7		3					

Prikaz 9x9 sudokua (lak, srednje težak, težak)

4 Pokretanje rešavača

Program se pokreće sa `./main` velicina_sudokua nivo, u tekstualnim fajlovima `ubcsat_*.txt` i `minisat.txt` se nalaze rezultati koje smo dobili pokretanjem `ubcsat` rešavača sa algoritmima `saps`, `gsat` i `novelty`, kao i `minisat` rešavač.

Naglašavamo da se na sledećoj slici može videti da svi rešavači i algoritmi imaju veliki broj parametara kojima se menja konfiguracija rešavača, a time i mogućnosti rešavanja, ali to prevazilazi potrebe ovog rada.



```
ubcsat_novelty.txt
149 100 0 26 46356 100000
150 # No Solution found for -target 0
151
152
153 Variables = 4096
154 Clauses = 117356
155 TotalLiterals = 241772
156 TotalCPULapsed = 12.410
157 FlipsPerSecond = 805802
158 RunsExecuted = 100
159 SuccessfulRuns = 0
160 PercentSuccess = 0.00
161 Steps_Mean = 100000
162 Steps_CoeffVariance = 0
163 Steps_Median = 100000
164 CPUTime_Mean = 0.1241
165 CPUTime_CoeffVariance = 0
166 CPUTime_Median = 0.1241
167

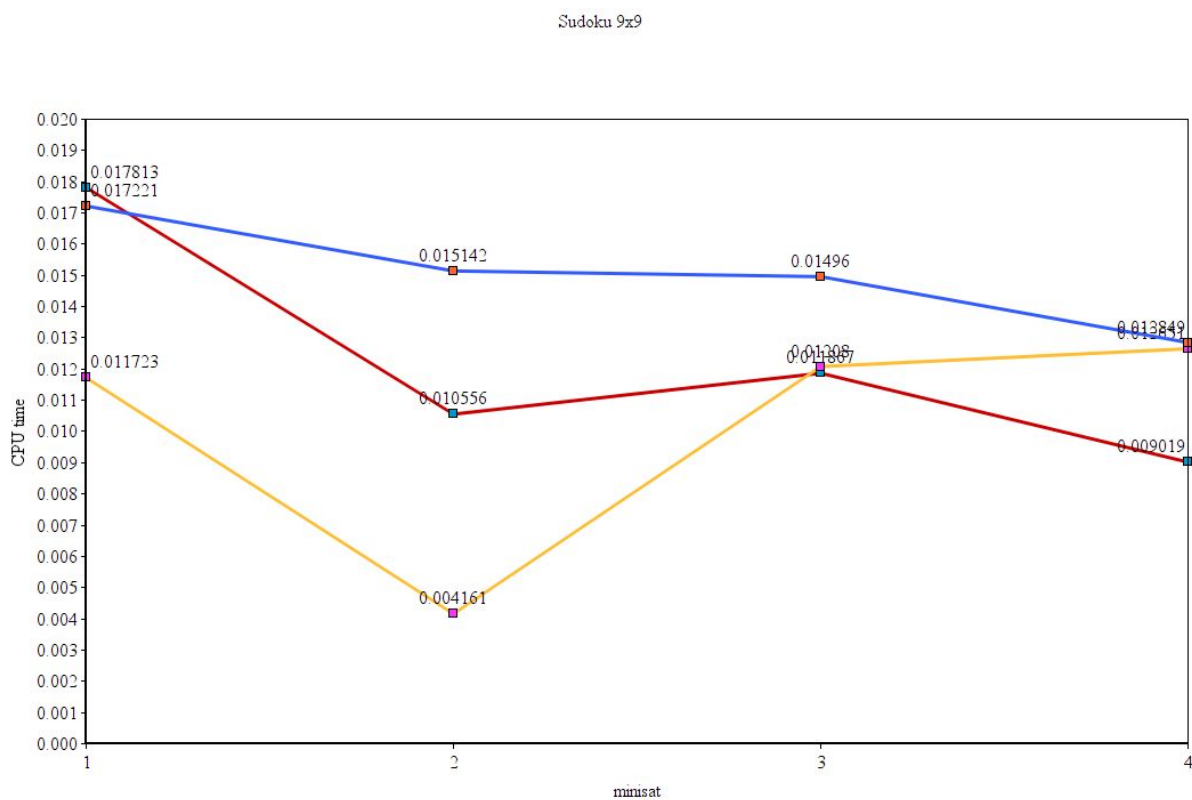
minisat.txt
2 =====[ Problem Statistics ]=====
3 |
4 | Number of variables: 4096
5 | Number of clauses: 117248
6 | Parse time: 0.20 s
7 | Eliminated clauses: 0.00 Mb
8 | Simplification time: 0.60 s
9 |
10 =====[ Search Statistics ]=====
11 | Conflicts | ORIGINAL | LEARNIT | Progress |
12 | | Vars | Clauses | Literals | Limit | Clauses Lit/Cl |
13 |=====|=====|=====|=====|=====|=====|
14 |
15 restarts : 1
16 conflicts : 13 (16 /sec)
17 decisions : 43 (0.00 % random) (53 /sec)
18 propagations : 5850 (7276 /sec)
19 conflict literals : 63 (5.97 % deleted)
20 Memory used : 25.26 MB
21 CPU time : 0.804 s
22
23 SATISFIABLE

ubcsat_saps.txt
53 1 0 6 10883 100000
54 # No Solution found for -target 0
55
56
57 Variables = 4096
58 Clauses = 117356
59 TotalLiterals = 241772
60 TotalCPULapsed = 0.160
61 FlipsPerSecond = 625000
62 RunsExecuted = 1
63 SuccessfulRuns = 0
64 PercentSuccess = 0.00
65 Steps_Mean = 100000
66 Steps_CoeffVariance = 0
67 Steps_Median = 100000
68 CPUTime_Mean = 0.16
69 CPUTime_CoeffVariance = 0
70 CPUTime_Median = 0.16
71

ubcsat_gsat.txt
47 #
48 1 0 124 2020 100000
49 # No Solution found for -target 0
50
51
52 Variables = 4096
53 Clauses = 117356
54 TotalLiterals = 241772
55 TotalCPULapsed = 0.090
56 FlipsPerSecond = 1111111
57 RunsExecuted = 1
58 SuccessfulRuns = 0
59 PercentSuccess = 0.00
60 Steps_Mean = 100000
61 Steps_CoeffVariance = 0
62 Steps_Median = 100000
63 CPUTime_Mean = 0.09
64 CPUTime_CoeffVariance = 0
65 CPUTime_Median = 0.09
66
67
68
```

Prikaz svih dobijenih parametara minisat i ubcsat rešavača, na primeru rešavanja 16x16 sudokua

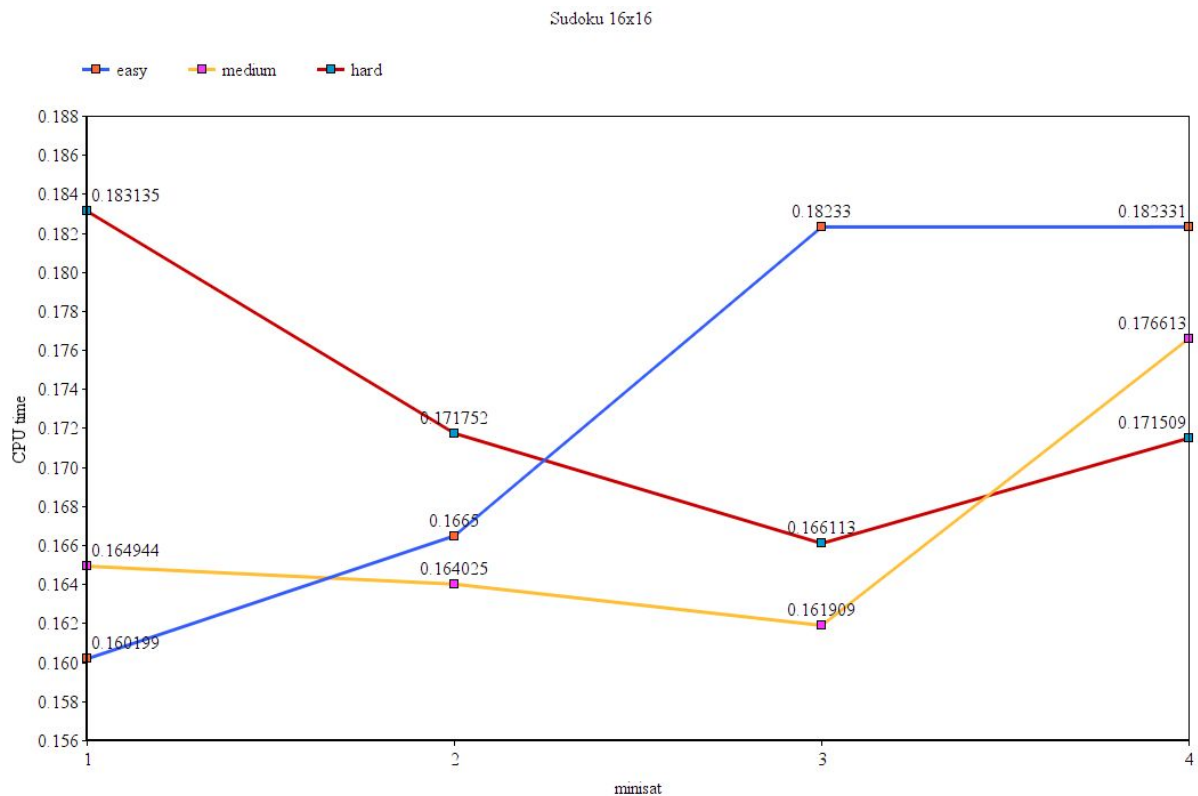
4.1 minisat



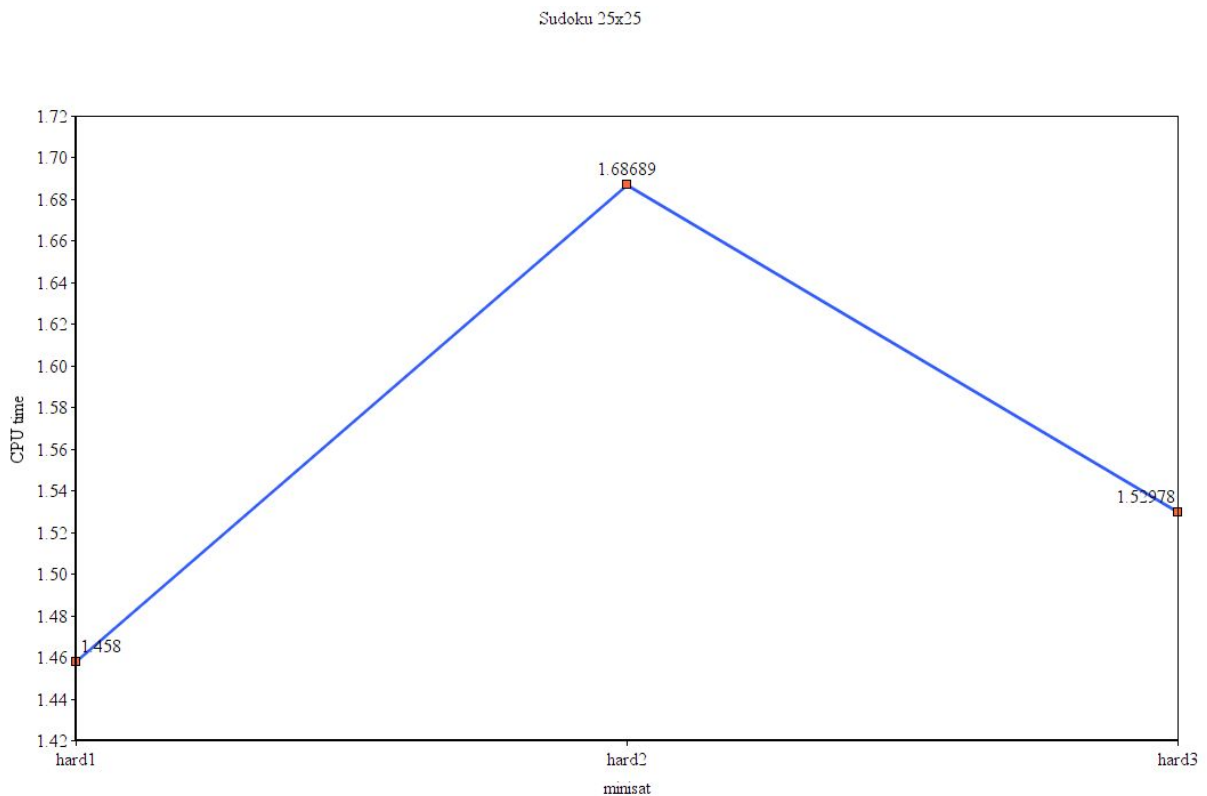
Na slici je prikazan rad minisat rešavača, primenjen na po 4 sudoku 9x9 instance različitih težina. Plavom linijom označen je laki nivo, žutom srednje teški nivo, a crvenom teški nivo.

Očekivali smo da vremena izvršavanja rastu sa povećanjem složenosti sudokua, međutim, ovi rezultati pokazuju drugačiju tendenciju, jer su instance uzimane sa različitih sajtova, pa se karakteristika složenosti ne može precizno odrediti.

Treba primetiti da su vremena izvršavanja instanci približno jednaka (vremena za većinu instanci se razlikuju tek na trećoj decimali), iako bi se sa grafikona moglo zaključiti suprotno.



Na ovom grafikonu prikazana su vremena izvršavanja za sudoku dimenzija 16x16. Instance su uzete sa istog sajta, a eksperiment je vršen kao i u prethodnom primeru, što se može primetiti u legendi. I u ovom primeru, minisat rešavač radi sa skoro jednakim vremenima. Može se primetiti da sudoku instance srednje teškog nivoa imaju najmanje srednje vreme izvršavanja u oba primera.



Na ovoj slici prikazan je rad rešavača na 3 teške sudoku instance dimenzija 25x25. I ovde je vreme izvršavanja skoro konstantno.

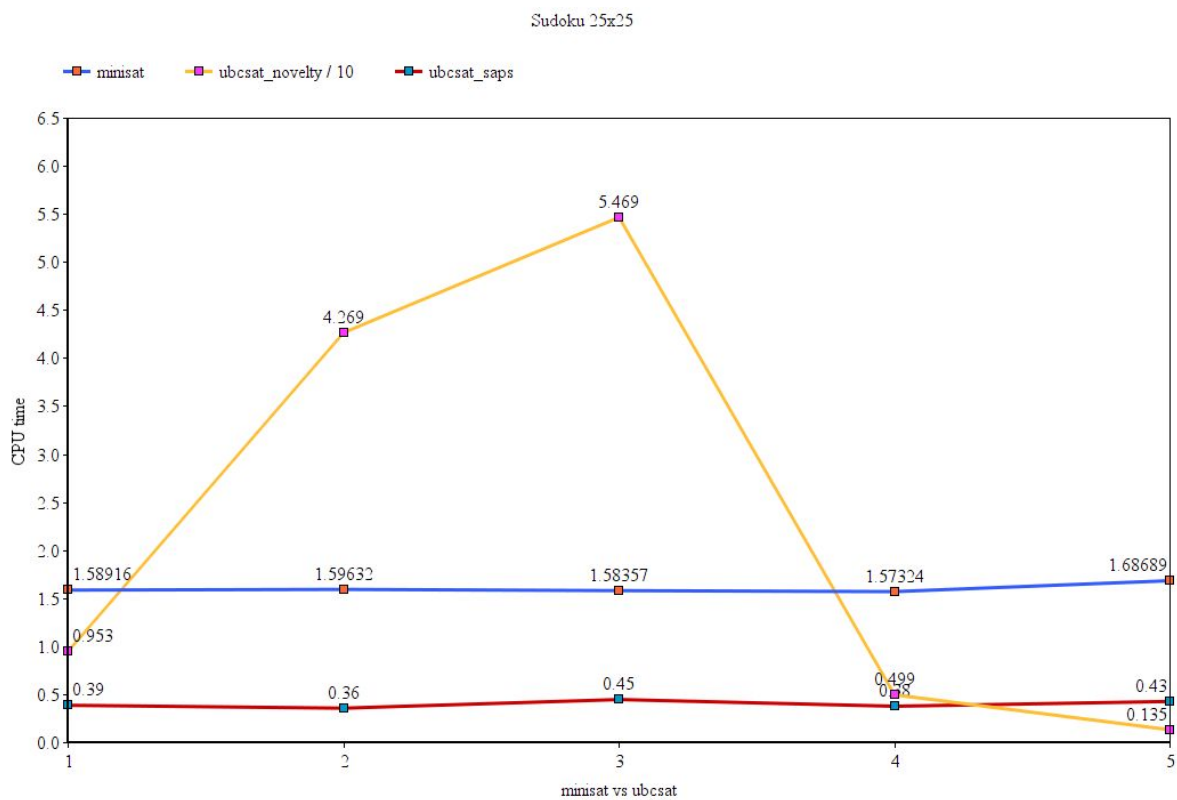
4.2 ubcsat

Postoje tri osnovna algoritma po kojima ubcsat rešavač radi i to su saps, gsat i novelty.

Pokretanjem rešavača pomoću gsat algoritma, za sve sudoku instance koje smo testirali u ovom radu, nijednom nije dobijeno rešenje, pa smatramo da je problem u samom gsat rešavaču. Stoga ćemo razmatrati rezultate za saps i novelty algoritme.

Pokretanjem rešavača pomoću ova dva algoritma, na sudoku instancama dimenzija 9x9, dobijene su približno iste vrednosti (razlika tek na trećoj decimali) za sva tri nivoa složenosti. Na nekoliko instanci, dobijeno procesorsko vreme izvršavanja iznosilo je 0.000, što ukazuje da ovi algoritmi rešavaju instance za vreme ispod 1 milisekunde. Stoga nisu pravljeni grafikoni vremena izvršavanja.

4.3 minisat vs ubcsat



Na grafikonu se mogu videti vremena izvršavanja za jednu sudoku instancu dimenzije 25x25 koju smo pokrenuli 5 puta. Na toj instanci su dobijeni rezultati za oba rešavača, pa smo nju analizirali više puta, jer ostale instance nisu davale rešenja za oba rešavača ili algoritma u slučaju ubcsat-a.

Važno je primetiti da algoritam novelty zahteva mnogo više vremena, pa je njegovo vreme smanjeno 10 puta zbog preglednosti. Minisat i ubcsat_saps imaju skoro konstantno vreme izvršavanja, dok je vreme izvršavanja ubcsat_novelty algoritma nestabilno.

5 Zaključak

- Uspešno su testirani rešavači
- Rešavali smo i manje dimenzije od prikazanih, kao i pravougaone sudoku tabele
- Gsat algoritam ubcsat rešavača nije davao rezultate
- Detaljnije upoznavanje sa parametrima bi sigurno dalo kvalitetnije rezultate