ΘΕΜΑ PROJECT

COMPILER FLAGS

ΤΡΑΧΑΝΗΣ ΔΗΜΗΤΡΗΣ 6108

ΤΡΑΧΑΝΗΣ ΓΕΩΡΓΙΟΣ 5789

Γενικά

- Οι compilers προσφέρουν μία σειρά από τεχνικές βελτιστοποίησης
- Στόχοι:
  - Αύξηση ταχύτητας εκτέλεσης
  - Μείωση μεγέθους
- Εφικτές γιατί υπάρχουν πολλαπλοί τρόποι να μεταφραστεί ένας αλγόριθμος σε κώδικα μηχανής.
  - Εξαρτώνται σε σημαντικό βαθμό από τα χαρακτηριστικά και το instruction set του επεξεργαστή

Είδη Βελτιστοποίησης(1/2)

- **Branch optimization**: Μείωση των διακλαδώσεων με συνδυασμό χωριστών κομματιών κώδικα.
- **Code motion:** Μεταβλητές βρόχου που δεν αλλάζει η τιμή τους μέσα στον βρόχο υπολογίζονται μία και μόνη φορά εκτός βρόχου.
- **Common subexpression elimination:** Όταν εκφράσεις έχουν μία κοινή υποέκφραση, η υποέκφρασή υπολογίζεται ξεχωριστά και το αποτέλεσμά της αντικαθίστανται σε κάθε έκφραση.
- **Constant propagation:** Οι σταθερές που χρησιμοποιούνται σε μια έκφραση συνδυάζονται, και παράγονται νέες.
- **Dead code elimination:** Αποβάλλει τον κώδικα που δεν χρησιμοποιείται ή δεν επηρεάζει το state του προγράμματος.
- **Dead store elimination:** Αποβάλλει εντολές αποθήκευσης όπου η τιμή που αποθηκεύεται δεν χρησιμοποιείται στη συνέχεια.

Είδη Βελτιστοποίησης(2/2)

- **Global register allocation:** Εναποθέτει μεταβλητές και εκφράσεις στους διαθέσιμους καταχωρητές με στόχο την διατήρηση όσων περισσότερων τελεστών στους καταχωρητές για γρηγορότερη εκτέλεση.
- **Inlining:** Αντικαθιστά τις κλήσεις συναρτήσεων με τον πραγματικό κώδικα τους, στο σημείο του προγράμματος από το οποίο καλούνται.
- **Instruction scheduling:** Ανακατατάζει εντολές για να ελαχιστοποιήσει το χρόνο εκτέλεσης.
- **Inter-procedural analysis:** Αποκαλύπτει συσχετίσεις στις κλήσεις συναρτήσεων, και αποβάλλει load, stores και υπολογιστμούς που δεν χρειάζονται.
- **Loop-unrolling:** Μειώνεται ο αριθμός των επαναλήψεων, γράφοντας πολλαπλά iterations στο σώμα του βρόγχου.
- **Store motion:** Βάζει εντολές αποθήκευσης έξω από επαναλήψεις.

Στόχος & εργαλεία εργασίας

- Στόχος:
  - Ο έλεγχος της απόδοσής ενός προγράμματος για διαφορετικές σημαιές βελτιστοποίησης.
- Εργαλεία:
  - Visual C++ compiler από το Visual Studio 2008
  - C και C++ benchmarks από τα SPEC CPU2006
  - VTune της Intel για τη μέτρηση της απόδοσης των προγραμμάτων σε κάθε είδος compiling.

Στόχος & εργαλεία εργασίας

- Τα χαρακτηριστικά του υπολογιστή στον οποίο θα γίνουν οι μετρήσεις είναι:

Processor	45nm Intel Core 2 Duo
L2 Cache	3MB
Frequency	2.26 GHz
Architecture	IA-32
Memory	2936MB
OS	Windows XP Professional SP3

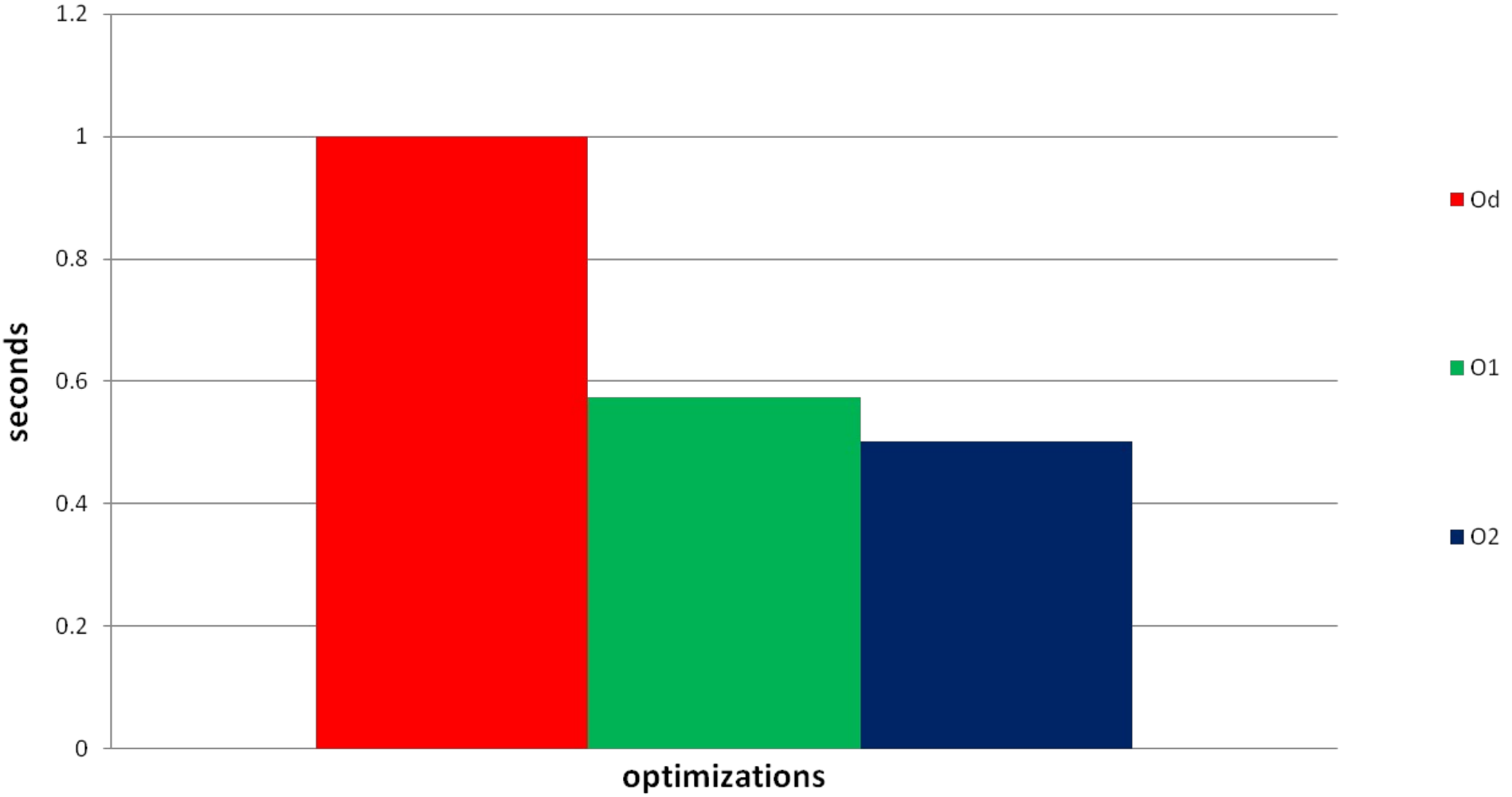
Η χρήση του Visual C++ compiler

- Οι τεχνικές βελτιστοποίησης του compiler επιλέγονται με την χρήση optimization flags:
  - /O1: ελαχιστοποίηση του μεγέθους (**/Og /Os /Oy /Ob2**)
  - /O2: μεγιστοποίηση της ταχύτητας εκτέλεσης (**Og /Oi /Ot /Oy /Ob2**)
  - /Ob: ελέγχει το inlining
  - /Od: απενεργοποιεί τις βελτιστοποιήσεις
  - /Og: common subexpression, global register allocation, code motion
  - /Os: προτίμηση για βελτιστοποιήσεις μεγέθους έναντι αυτών για ταχύτητα
  - /Ot: προτίμηση για βελτιστοποιήσεις ταχύτητας έναντι αυτών για μέγεθος
  - /Ox: πλήρη βελτιστοποίηση
  - /Oy: αποτρέπει τη δημιουργία frame pointers στη στοίβα κλήσεων
- Εμείς θα πειραματιστούμε με τα O1, O2, Od και με SSE2 εντολών

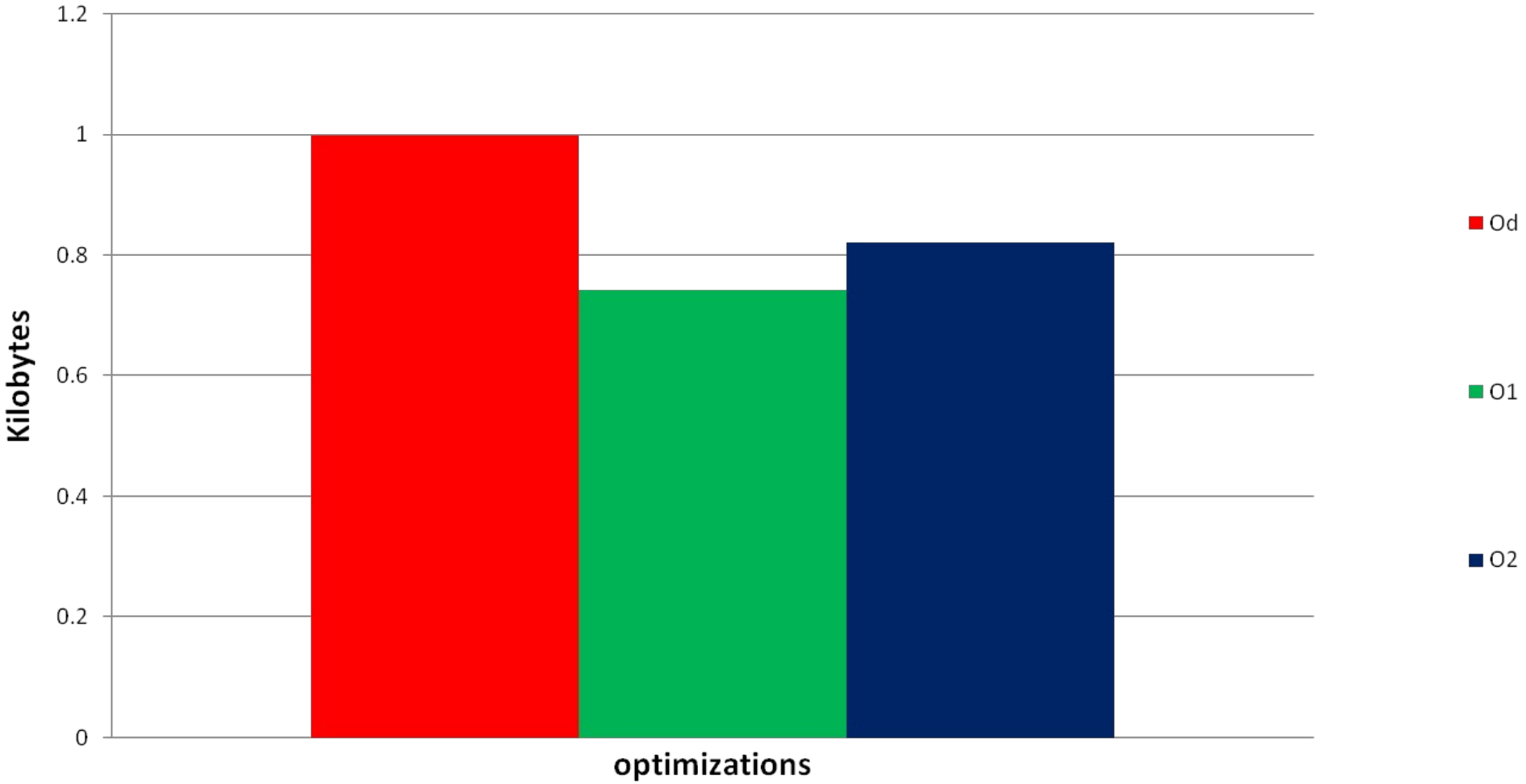
Το VTune της intel

- Το Vtune της Intel θα το χρησιμοποιήσουμε για να μετρήσουμε την απόδοση των εκτελέσιμων αρχείων σε κάθε βελτιστοποίηση με την οποία έχουν γίνει compile.
- Οι τιμές που θα πάρουμε είναι:
  - **CPI** (clocks per instruction)
  - **Instructions retired**
  - **L1 cache misses**
  - **L2 cache misses**
  - **Execution time**
  - **Size of exec file**

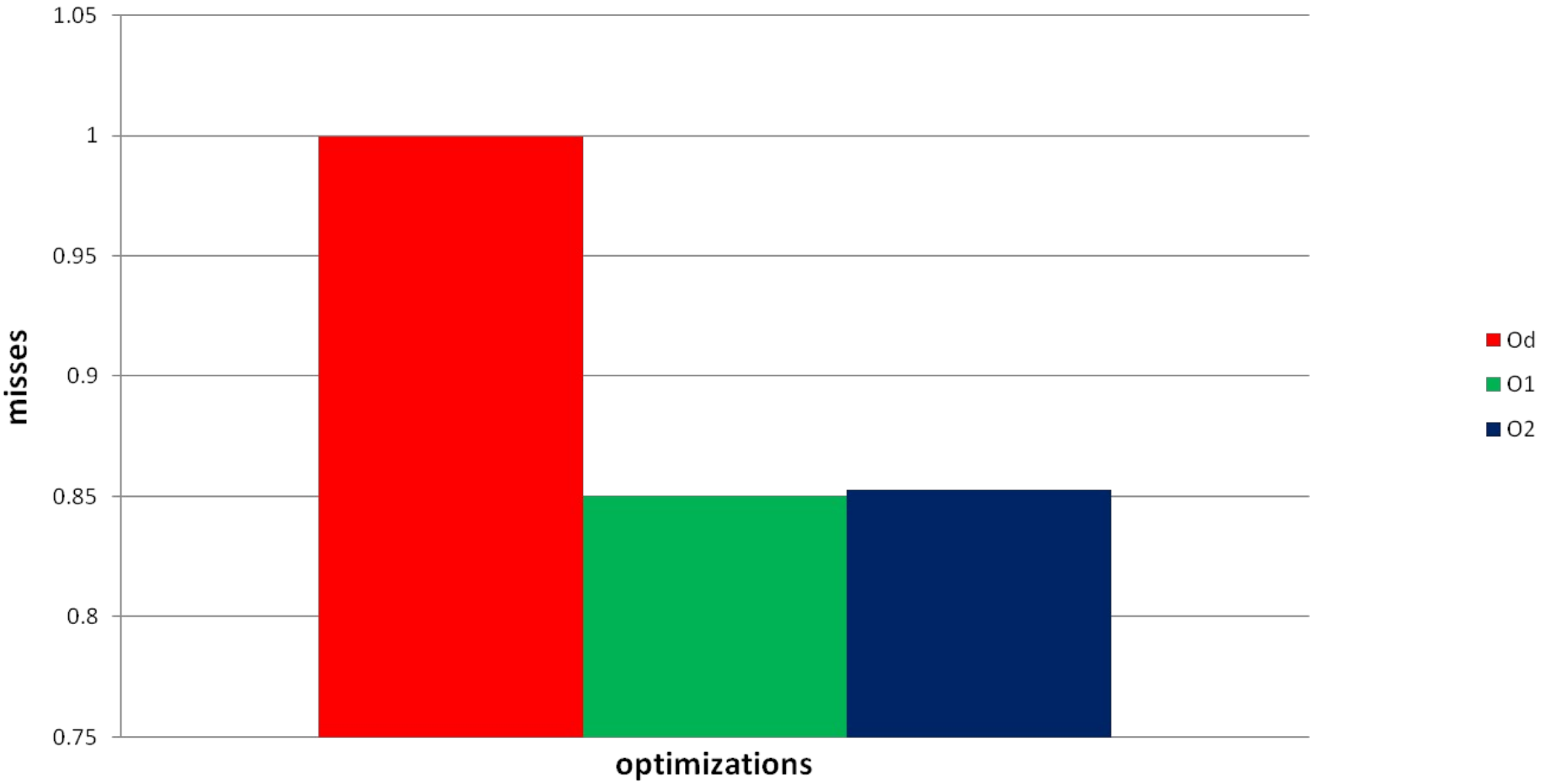
Κανονικοποιημένος Χρόνος Εκτέλεσης



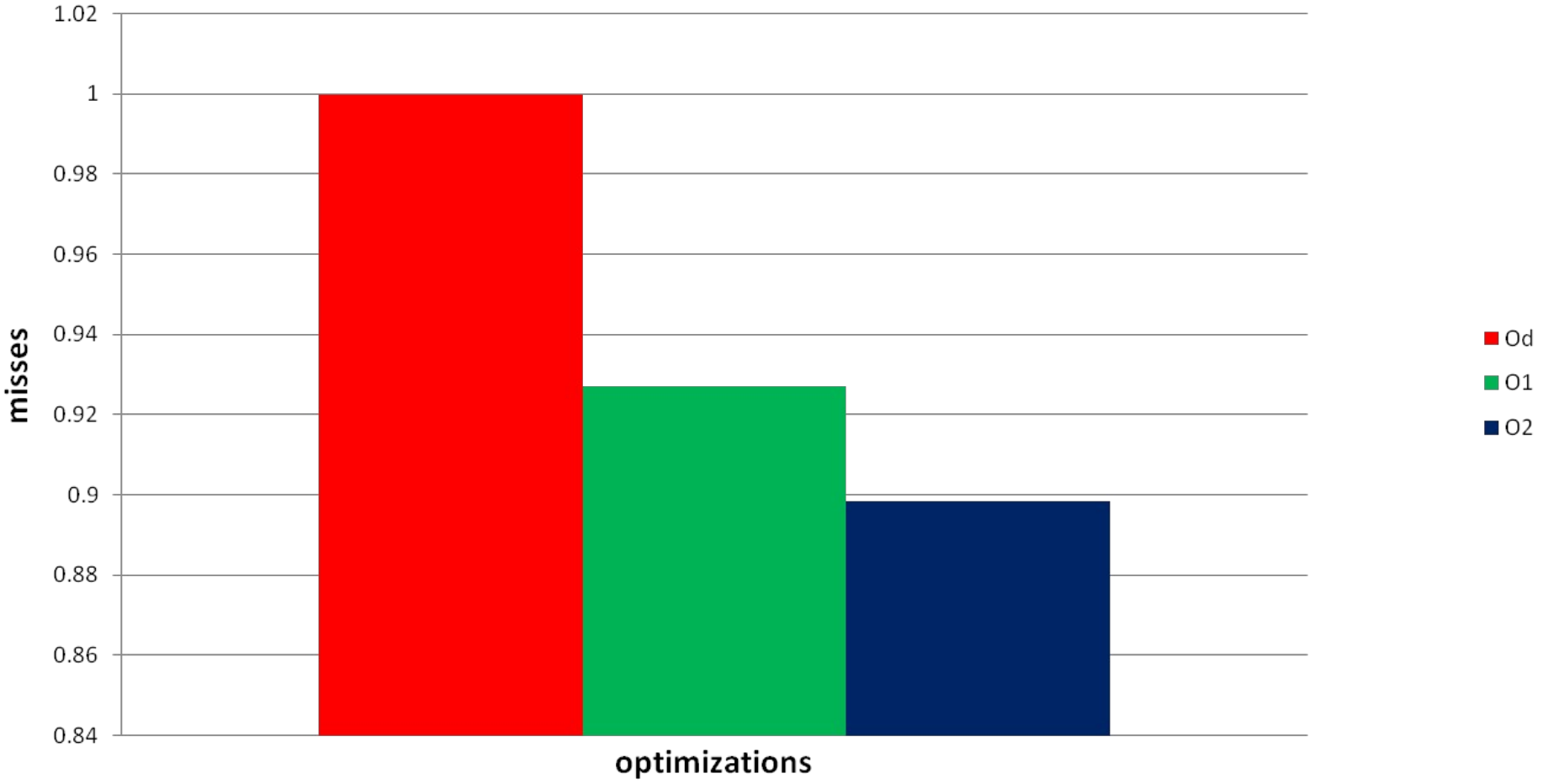
Κανονικοποιημένο Μέγεθος Εκτελέσιμου



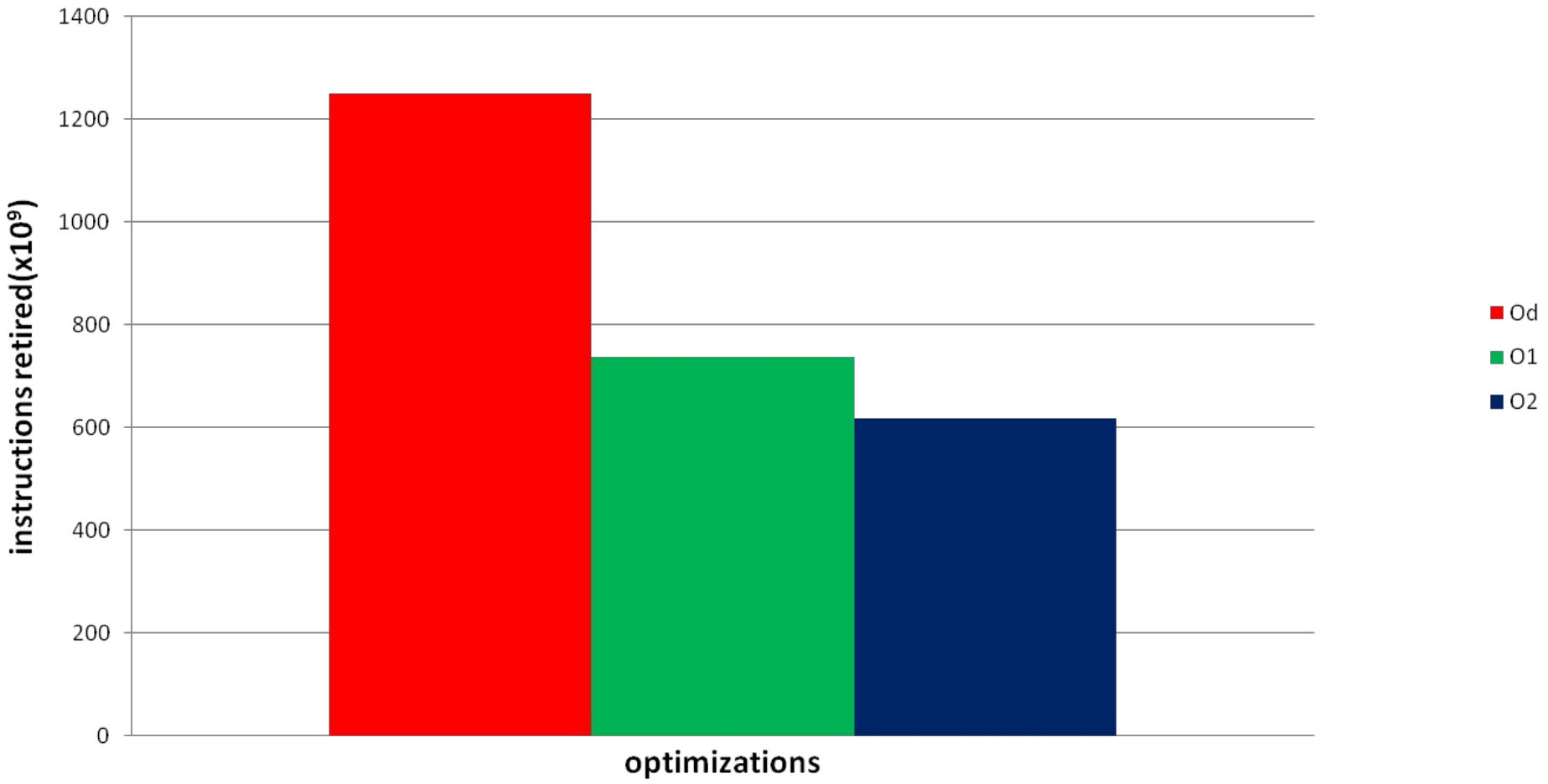
Κανονικοποιημένες Αστοχίες L1



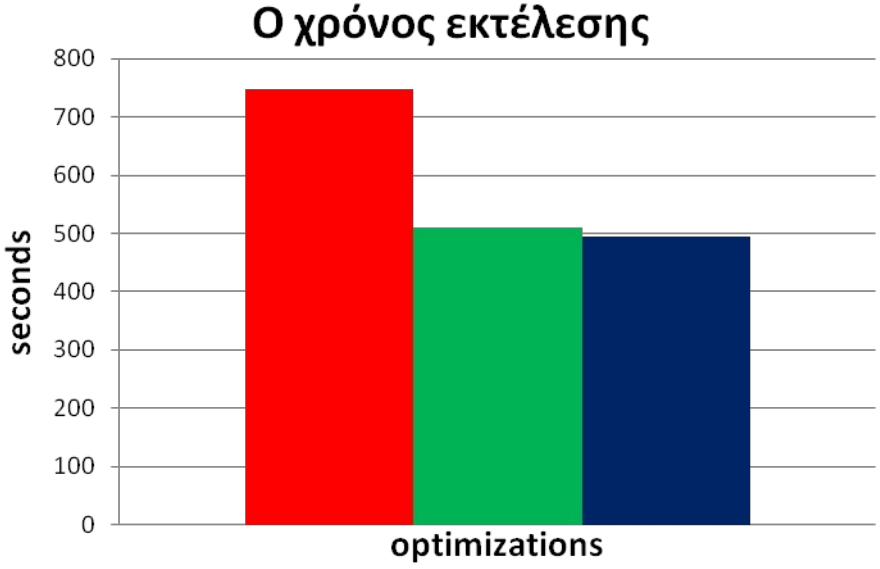
Κανονικοποιημένες Αστοχίες L2

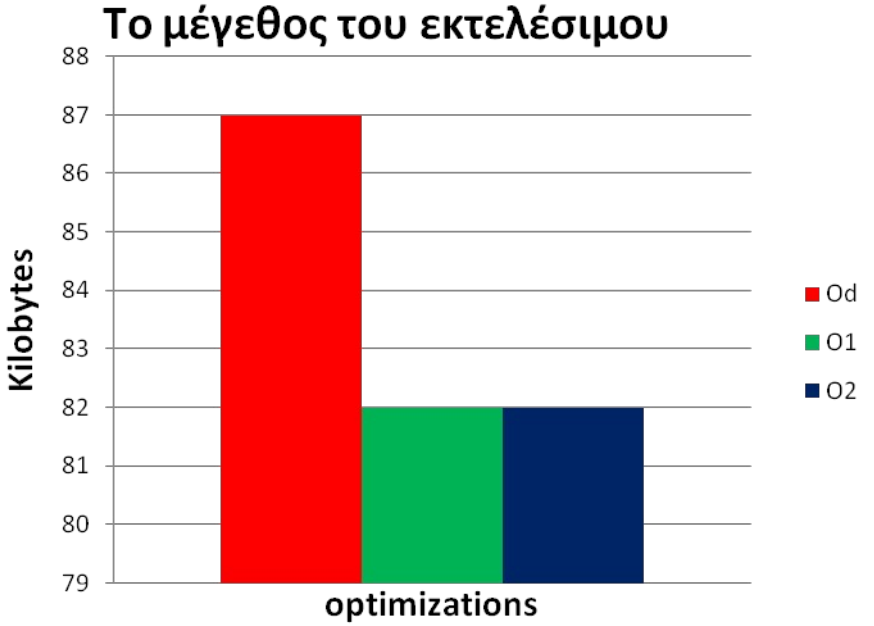


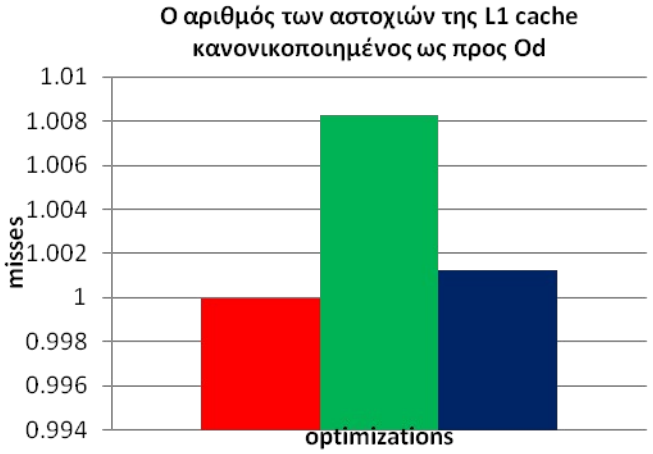
Γεωμετρικός Μέσος Εντολών που εκτελέστηκαν

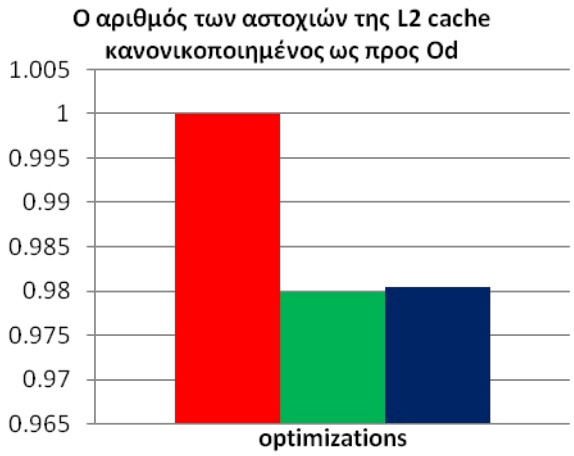


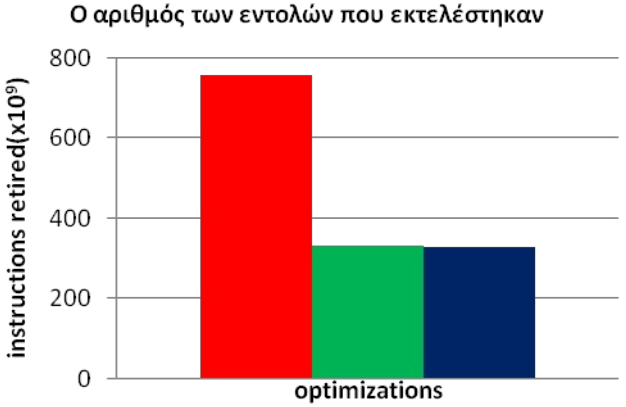
Αποτελέσματα Μετρήσεων για το benchmark 429.mcf



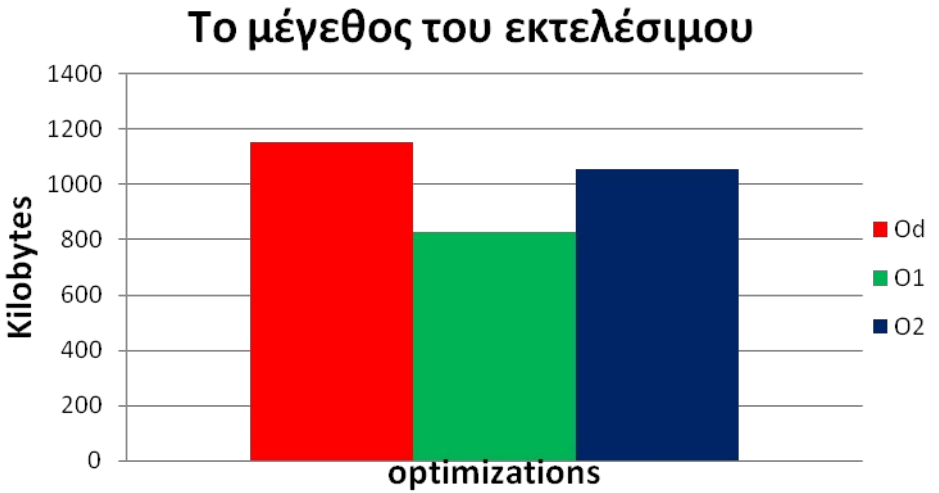


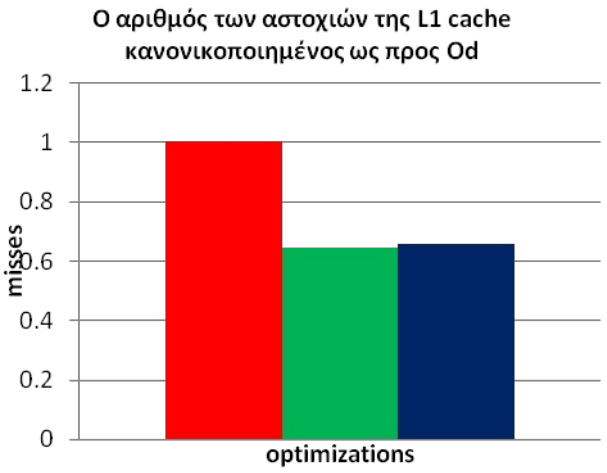


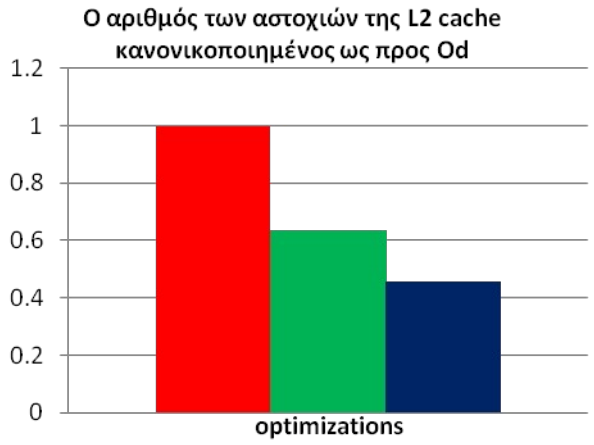


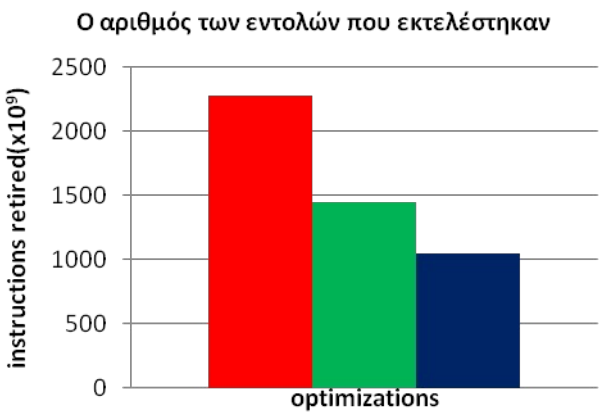


Αποτελέσματα Μετρήσεων για το benchmark 453.povray









Σχολιασμός αποτελεσμάτων

- Όπως φαίνεται από τα αποτελέσματα των μετρήσεων, αυτά έρχονται να επαληθεύσουν τα θεωρητικά αναμενόμενα αποτελέσματα. Με τη βελτιστοποίηση /O2 έχουμε το ταχύτερο εκτελέσιμο, ενώ με τη βελτιστοποίηση /O1 το μικρότερο σε μέγεθος.

- Χωρίς βελτιστοποιήσεις (/Od)έχουμε τη χειρότερη απόδοση, τόσο σε ταχύτητα και μέγεθος εκτελέσιμου, όσο και σε αστοχίες στις κρυφές μνήμες.

Παράρτημα

Αποτελέσματα Μετρήσεων

	Od						
benchmarks	CPI	IPC	instr_ret(x10^9)	L1$ misses(x10^6)	L2$ misses(x10^6)	execution time(s)	exe file size(KB)
400.perlbench	0,866	1,154734	1671	4579	1554	642,593	1308
401.bzip2	0,884	1,131222	204	1358	112	78,656	146
429.mcf	2,282	0,438212	755	51254	14349	746,921	87
444.namd	1,697	0,589275	1175	37853	92	856,484	572
445.gobmk	1,169	0,855432	350	1129	142	176,734	3340
450.soplex	1,368	0,730994	1197	20489	12504	705,296	448
453.povray	1,051	0,951475	2279	20506	33	1039,69	1151
456.hmmer	1,034	0,967118	2023	5538	802	891,39	283
458.sjeng	1,096	0,912409	3615	8740	2342	1692,28	215
464.h264ref	0,744	1,344086	1124	1811	342	357,109	738
470.lbm	1,282	0,780031	2190	65051	37292	1219,2	110
471.omnetpp	1,307	0,765111	2065	18637	17410	1165,02	930
473.astar	1,041	0,960615	1720	14714	2077	765,89	127

Παράρτημα

Αποτελέσματα Μετρήσεων

	O1						
benchmarks	CPI	IPC	instr_ret(x10^9)	L1$ misses(x10^6)	L2$ misses(x10^6)	execution time(s)	exe file size(KB)
400.perlbench	0,744	1,344086	1147	4374	1499	387,625	815
401.bzip2	0,871	1,148106	103	1373	103	38,625	106
429.mcf	3,512	0,284738	331	51676	14060	510,906	82
444.namd	0,722	1,385042	721	6136	68	222,937	344
445.gobmk	1,085	0,921659	239	1161	158	112,187	2995
450.soplex	1,676	0,596659	735	20390	12006	532,218	315
453.povray	0,894	1,118568	1445	13176	20580	567,453	823
456.hmmer	1,045	0,956938	1200	5534	732	534,531	214
458.sjeng	0,937	1,067236	2462	8741	2316	984,687	162
464.h264ref	0,636	1,572327	648	1805	270	176,578	468
470.lbm	2,039	0,490436	1300	64711	37254	1155,28	95
471.omnetpp	2,007	0,498256	1584	20334	18598	692,375	595
473.astar	1,41	0,70922	690	15998	2338	417,515	115

Παράρτημα

Αποτελέσματα Μετρήσεων

	O2						
benchmarks	CPI	IPC	instr_ret(x10^9)	L1$ misses(x10^6)	L2$ misses(x10^6)	execution time(s)	exe file size(KB)
400.perlbench	0,724	1,381215	1152	4402	1503	383,203	956
401.bzip2	0,831	1,203369	100	1371	119	36,14	111
429.mcf	3,447	0,290107	327	51318	14067	494,015	82
444.namd	0,677	1,477105	719	6029	67	208,453	378
445.gobmk	1,046	0,956023	221	1168	157	100,312	3116
450.soplex	2,129	0,469704	364	19393	11848	338,187	394
453.povray	0,791	1,264223	1053	13424	15	372,312	1051
456.hmmer	0,978	1,022495	1104	5459	696	460,296	231
458.sjeng	0,883	1,132503	2235	9407	2306	844,625	176
464.h264ref	0,646	1,547988	580	1812	266	160,484	542
470.lbm	2,002	0,4995	1293	64686	37189	1129,92	92
471.omnetpp	2,157	0,463607	664	20563	18495	626,703	707
473.astar	1,343	0,744602	702	16058	2100	404,687	122