Προηγμένα Θέματα Αρχιτεκτονικής Υπολογιστών
1η άσκηση

Νικόλαος Παγώνας, el18175

0 Εισαγωγή

Στα πλαίσια της 1ης άσκησης θα χρησιμοποιήσουμε το PIN και τα benchmarks:

- blackscholes
- bodytrack
- canneal
- fluidanimate
- freqmine
- ecluster
- swaptions

προκειμένου να εξετάσουμε πώς επηρεάζεται η επίδοση (IPC και cache misses) της ιεραρχίας μνήμης που περιγράφεται στην εκφώνηση, όταν μεταβάλλουμε τις παραμέτρους της L1 cache, της L2 cache και του TLB. Αρχικά θα κάνουμε την υπόθεση ότι ο κύκλος ρολογιού είναι σταθερός και στη συνέχεια θα υποθέσουμε ότι με τη μεταβολή των χαρακτηριστικών υπεισέρχεται και κάποιο overhead στον κύκλο ρολογιού, κάτι που είναι και πιο κοντά στην πραγματικότητα. Κάθε παράμετρος χαρακτηρίζεται ως:

- *ωφέλιμη* αν με αύξησή της έχουμε καλύτερα αποτελέσματα,
- *ζημιογόνα*, αν έχουμε χειρότερα,
- *ουδέτερη*, αν δεν έχουμε σημαντικές αλλαγές.

Τέλος, σε γενικές γραμμές (και για να μην επαναλαμβανόμαστε σε κάθε γραφική παράσταση), επισημαίνουμε τα εξής:

- Αναμένουμε το μέγεθος της cache/του TLB να παίζει σημαντικότερο ρόλο όταν τα benchmarks προκαλούν πολλά capacity misses
- Αντίστοιχα, αναμένουμε το associativity να παίζει σημαντικότερο ρόλο όταν τα benchmarks προκαλούν πολλά conflict misses
- Τέλος, αναμένουμε το block size να παίζει σημαντικότερο ρόλο όταν τα benchmarks προκαλούν πολλά compulsory misses

1 Ζητούμενο 1ο

Σε αυτό το ζητούμενο θεωρούμε ότι ο κύκλος ρολογιού είναι σταθερός. Μεταβάλλουμε τα χαρακτηριστικά μίας μνήμης τη φορά και καταγράφουμε τα αποτελέσματα.

1.1 L1 cache

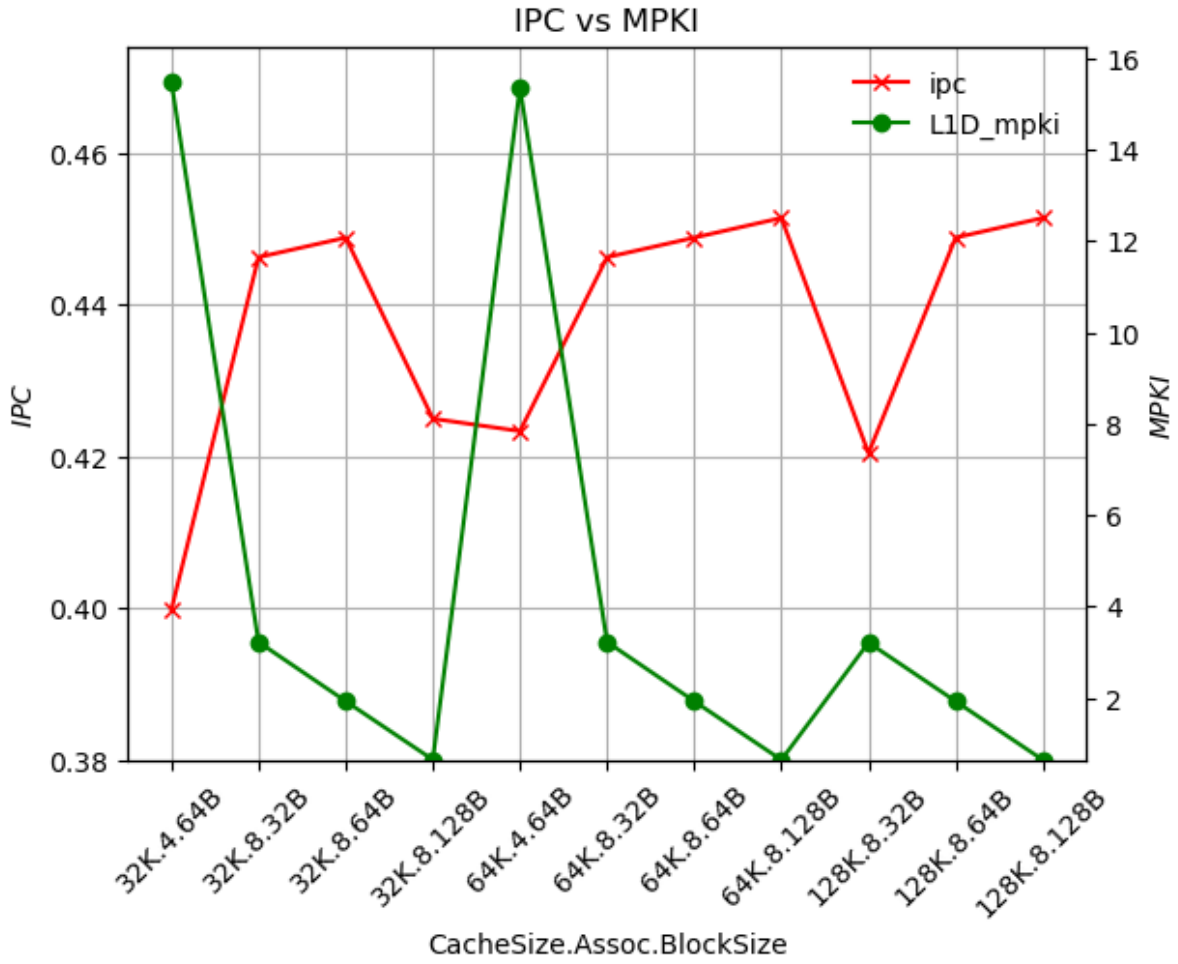
Σε αυτό το κομμάτι οι παράμετροι της L2 cache και του TLB διατηρούνται σταθερές και ίσες με:

- L2 size: 1024 KB
- L2 associativity: 8
- L2 block size: 128 B
- TLB size (entries): 64
- TLB associativity: 4
- TLB page size: 4096 B

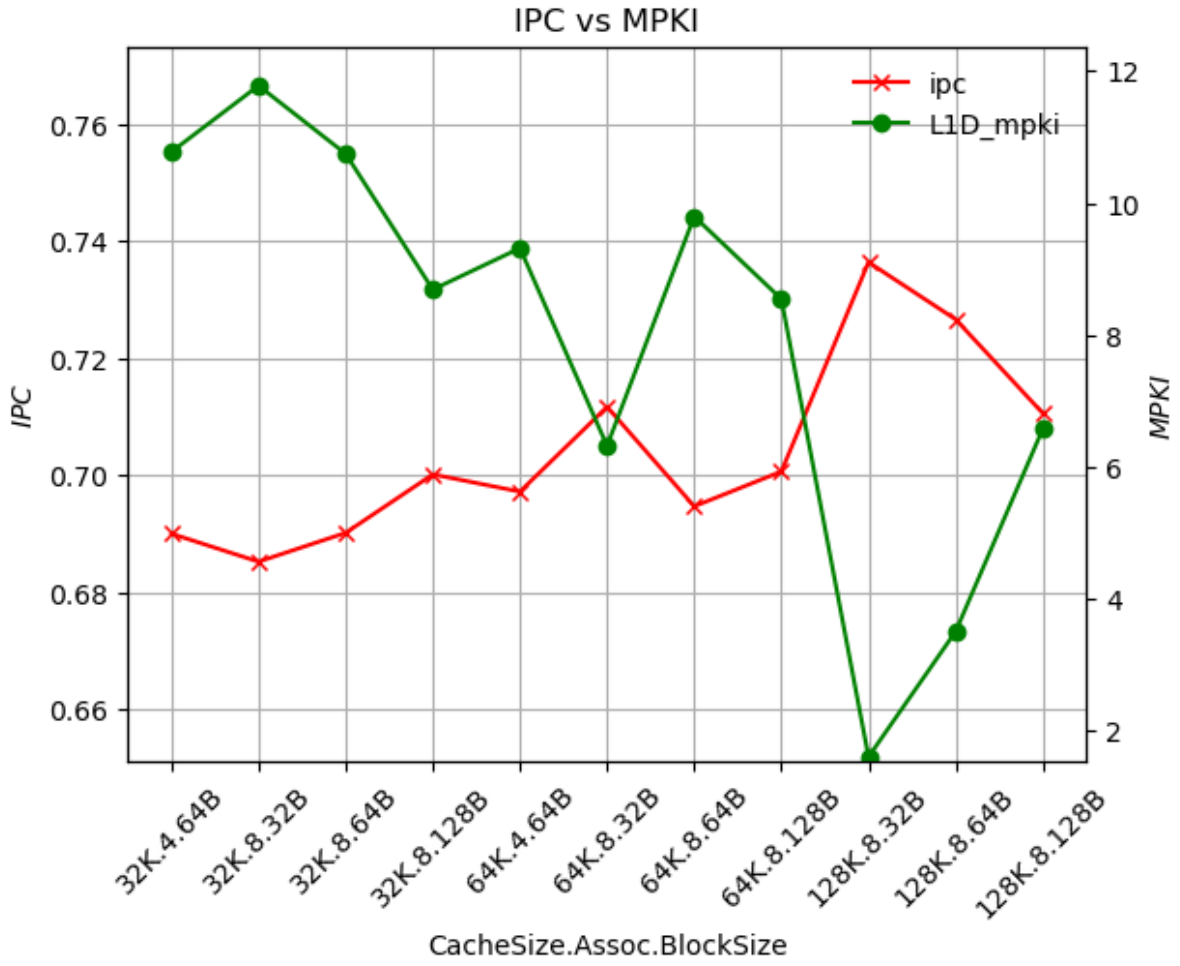
Όσον αφορά τα χαρακτηριστικά της L1 cache, έχουμε τις εξής περιπτώσεις:

L1 size (KB)	L1 associativity	L1 block size (B)
32	4	64
32	8	32
32	8	64
32	8	128
64	4	64
64	8	32
64	8	64
64	8	128
128	8	32
128	8	64
128	8	128

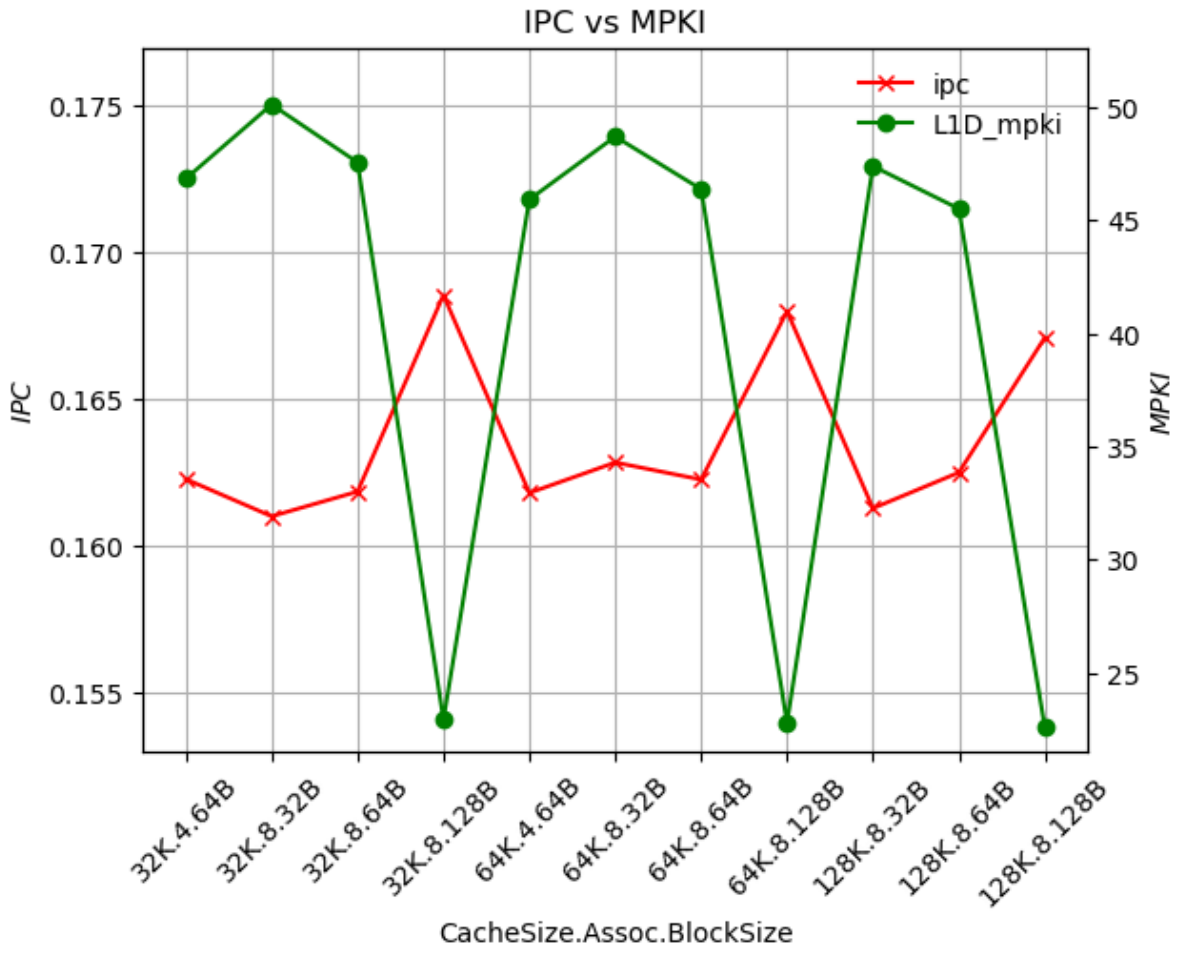
1.1.1 blackscholes



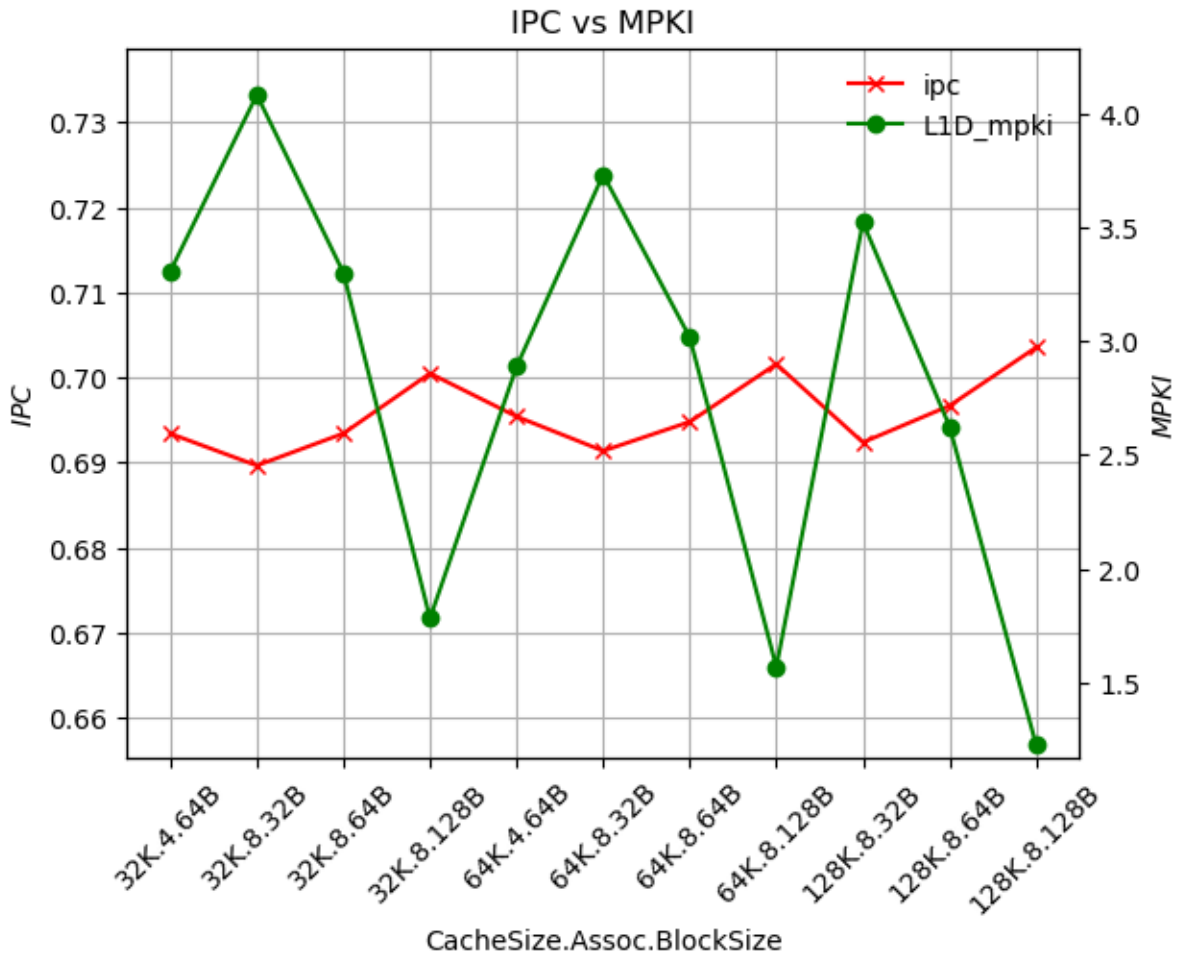
1.1.2 bodytrack



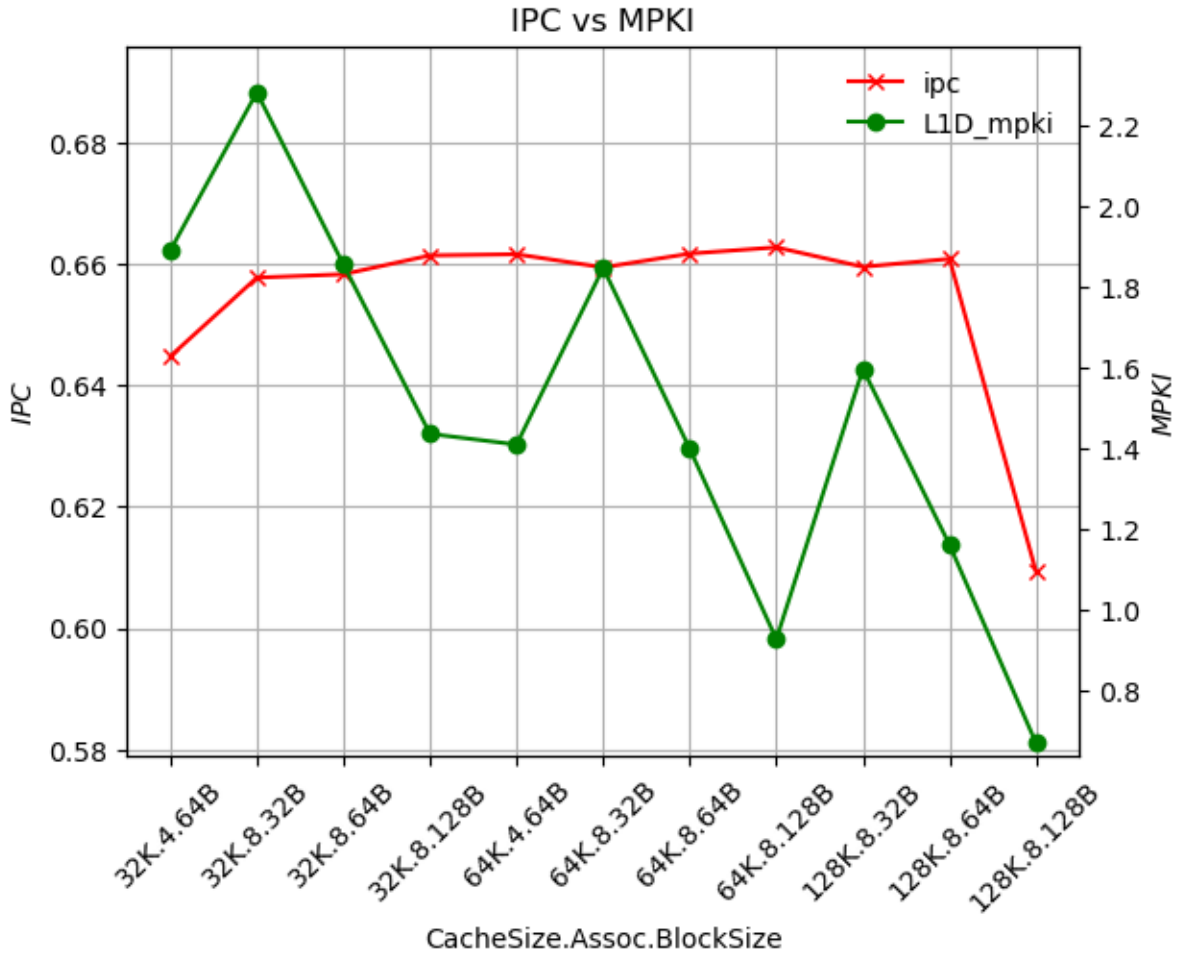
1.1.3 canneal


IPC vs MPKI
ipc
L1D_mpki
IPC
MPKI
0.175
0.170
0.165
0.160
0.155
50
45
40
35
30
25
32K.4.64B
32K.8.32B
32K.8.64B
32K.8.128B
64K.4.64B
64K.8.32B
64K.8.64B
64K.8.128B
128K.8.32B
128K.8.64B
128K.8.128B
CacheSize.Assoc.BlockSize

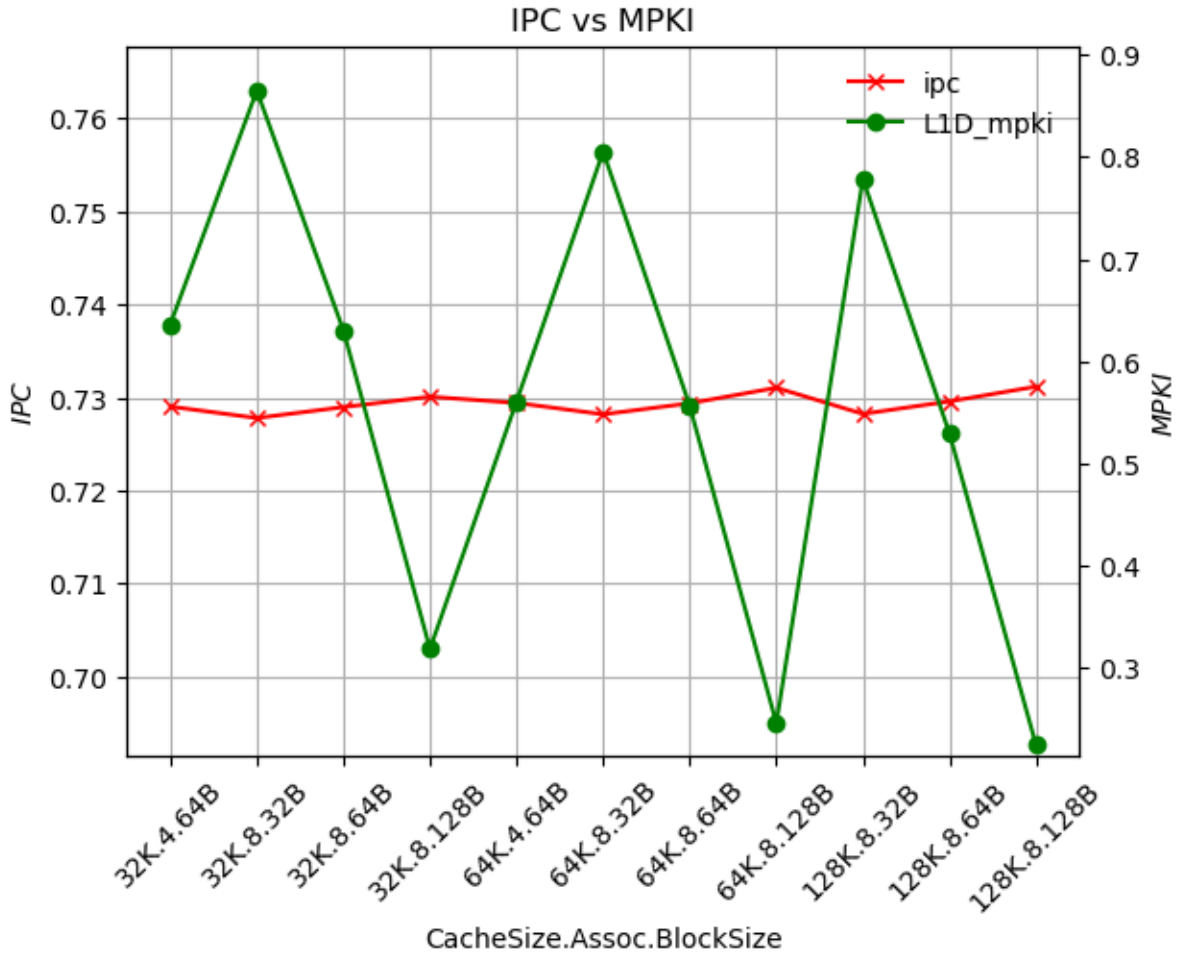
1.1.4 fluidanimate



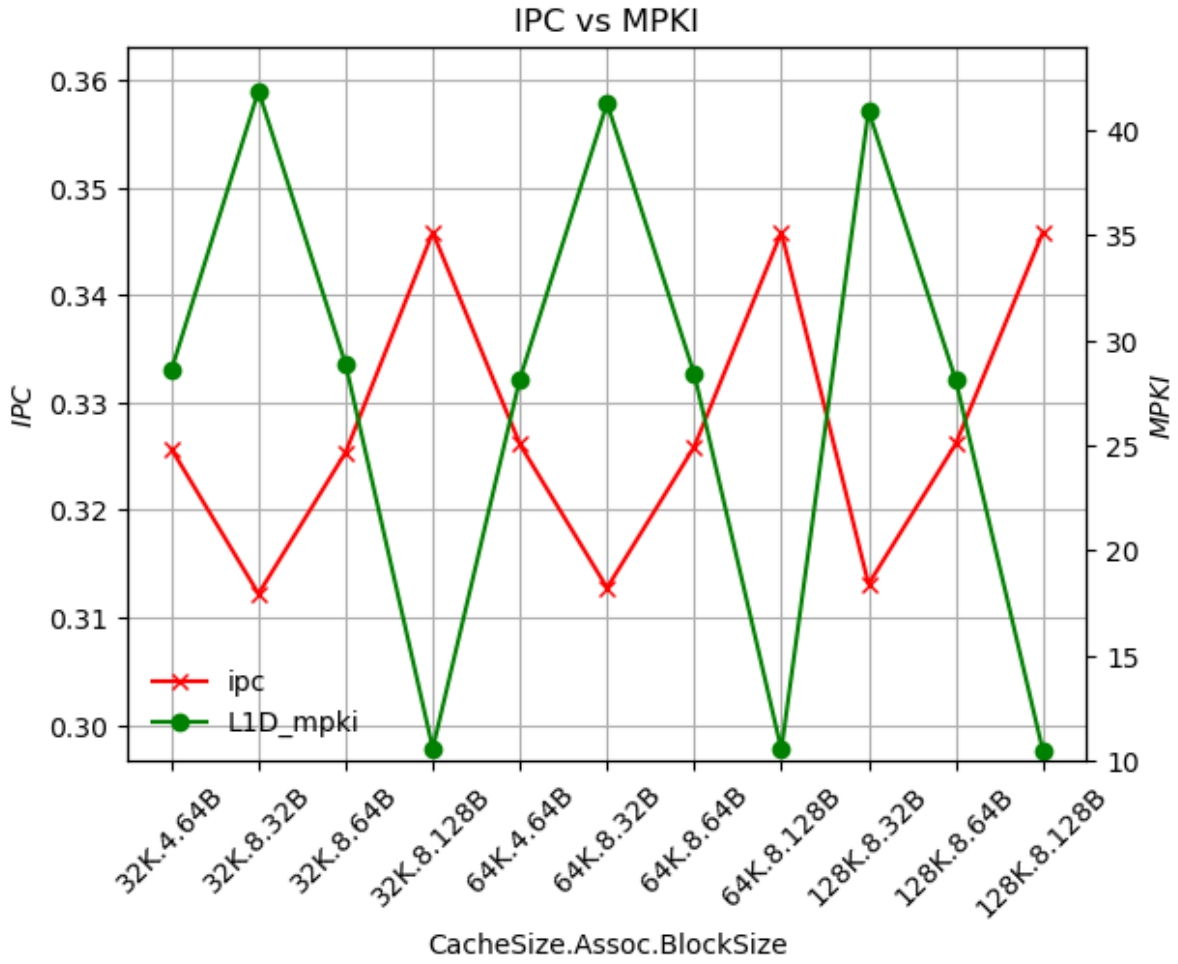
1.1.5 freqmine



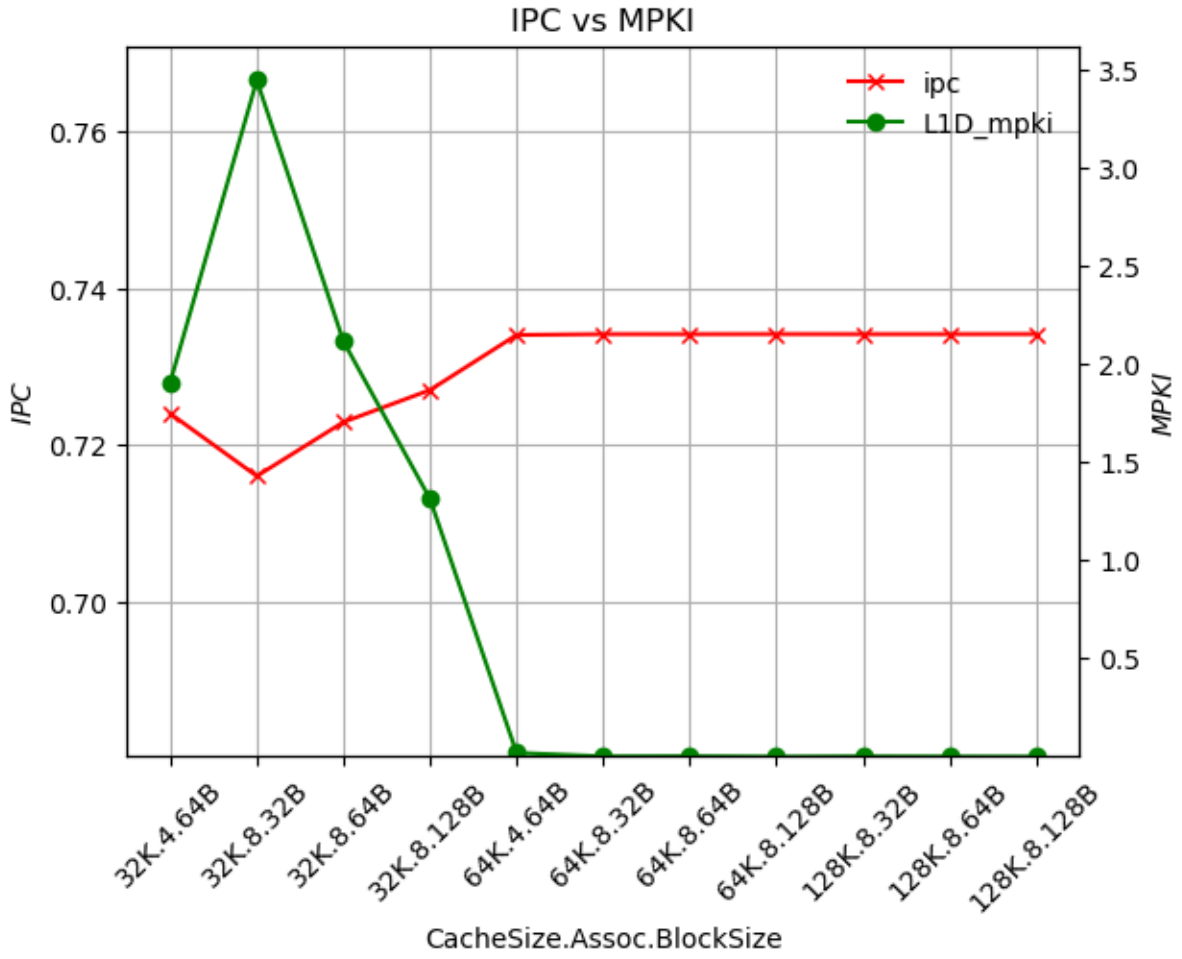
1.1.6 rtview


IPC vs MPKI
ipc
L1D_mpki
IPC
MPKI
0.76
0.75
0.74
0.73
0.72
0.71
0.70
0.9
0.8
0.7
0.6
0.5
0.4
0.3
32K.4.64B
32K.8.32B
32K.8.64B
32K.8.128B
64K.4.64B
64K.8.32B
64K.8.64B
64K.8.128B
128K.8.32B
128K.8.64B
128K.8.128B
CacheSize.Assoc.BlockSize

1.1.7 streamcluster

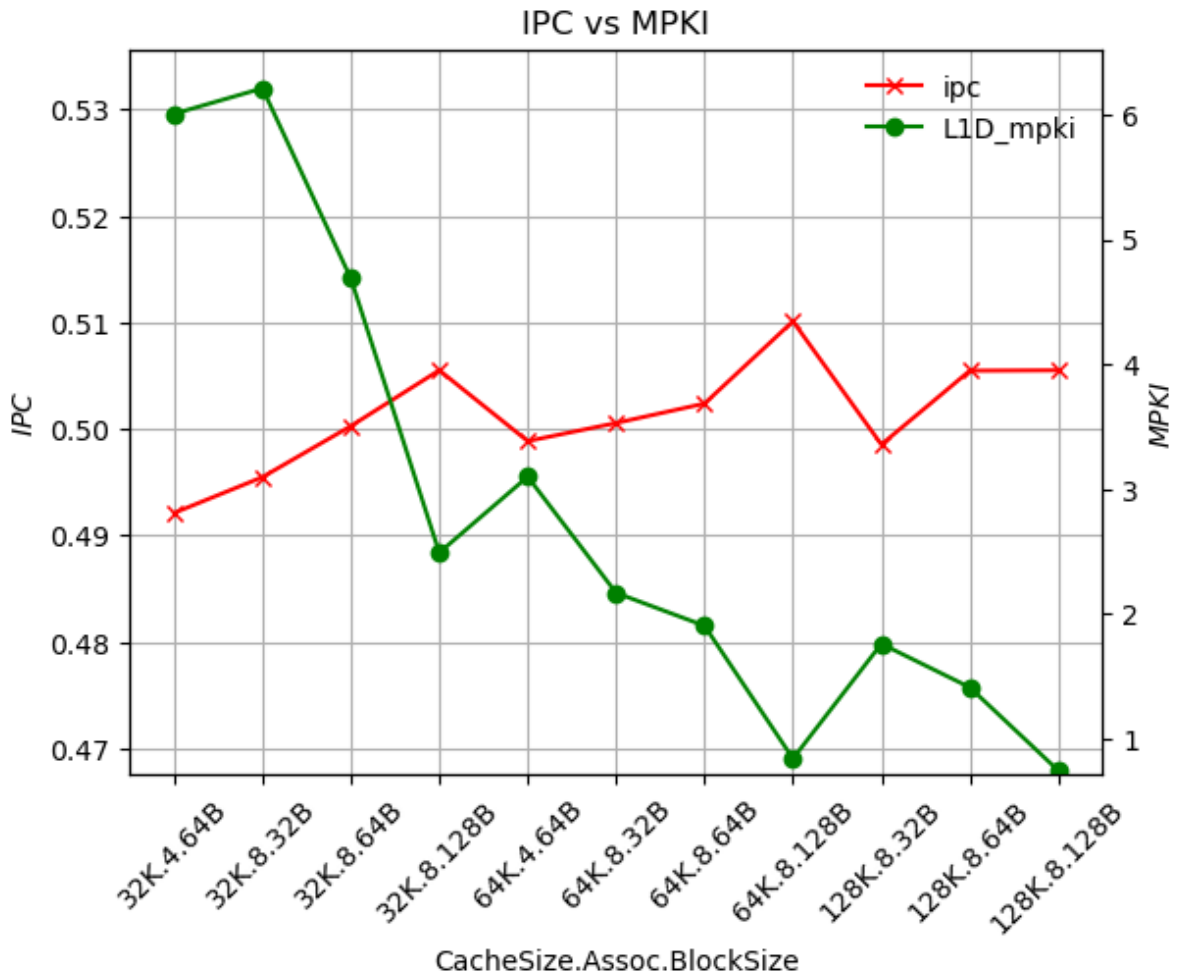


1.1.8 swaptions



Τέλος, απεικονίζουμε το πώς μεταβάλλεται ο γεωμετρικός μέσος όρος όλων των benchmark, όσο μεταβάλλονται τα χαρακτηριστικά της cache. Ο γεωμετρικός όρος συνηθίζεται στα benchmarks, το οποίο είναι λογικό, αφού οι έννοιες του speedup και του slowdown είναι πολλαπλασιαστικής φύσεως. Για παράδειγμα, ένα speedup $\times 2$ και ένα speedup $\times 0.5$ (δηλαδή slowdown $\times 2$) έχουν γεωμετρικό μέσο ίσο με $\sqrt{2 \times 0.5} = 1$ (καθόλου συνολικό speedup), ενώ έχουν αριθμητικό μέσο $(1+0.5)/2 = 0.75$, το οποίο δεν είναι αντιπροσωπευτικό αποτέλεσμα με βάση τη διαίσθησή μας.

1.1.9 Γεωμετρικός μέσος των benchmarks



1.1.10 Γενικές παρατηρήσεις για την L1

- Όπως είναι λογικό τα IPC και MPKI μεταβάλλονται με αντίστροφο τρόπο (όταν το ένα αυξάνει, το άλλο μειώνεται), αφού όσο πιο πολλά misses παρατηρούνται, τόσο περισσότεροι κύκλοι απαιτούνται για να εκτελεστεί η εντολή που τα προκάλεσε.
- Καλύτερη επιλογή φαίνεται να είναι η τριπλέτα:

  `(cache size, associativity, block size) = (64K, 8, 128B)`

- Το L1 cache size δείχνει να βελτιώνει σημαντικά την επίδοση στο swaptions, όταν αλλάζει από 32K σε 64K.
- Στα benchmarks bodytrack, canneal, fluidanimate, streamcluster και swaptions παρατηρούμε ότι η αύξηση του block size μειώνει το MPKI (compulsory misses) και άρα αυξάνει το IPC (σε άλλα benchmarks περισσότερο και σε άλλα λιγότερο).
- Το rtview επηρεάζεται λιγότερο σε σχέση με τα άλλα (IPC 0.70-0.71 για όλες τις τιμές των παραμέτρων).

- Σε όλα τα benchmarks εκτός από το swaptions, το μέγεθος της cache δεν παίζει σημαντικό ρόλο (πιθανόν να μην υπάρχουν αρκετά capacity misses ώστε το μέγεθος της cache να μπορεί να συμβάλλει).
- Στο blackscholes, όταν το associativity αυξήθηκε, αντίστοιχα αυξήθηκε και το IPC (άρα το πιθανότερο είναι ότι ελαττώθηκαν τα conflict misses).
- Τελικά οι παράμετροι που παίζουν μεγαλύτερο ρόλο είναι το associativity και το block size, και όχι τόσο το cache size.

1.2 L2 cache

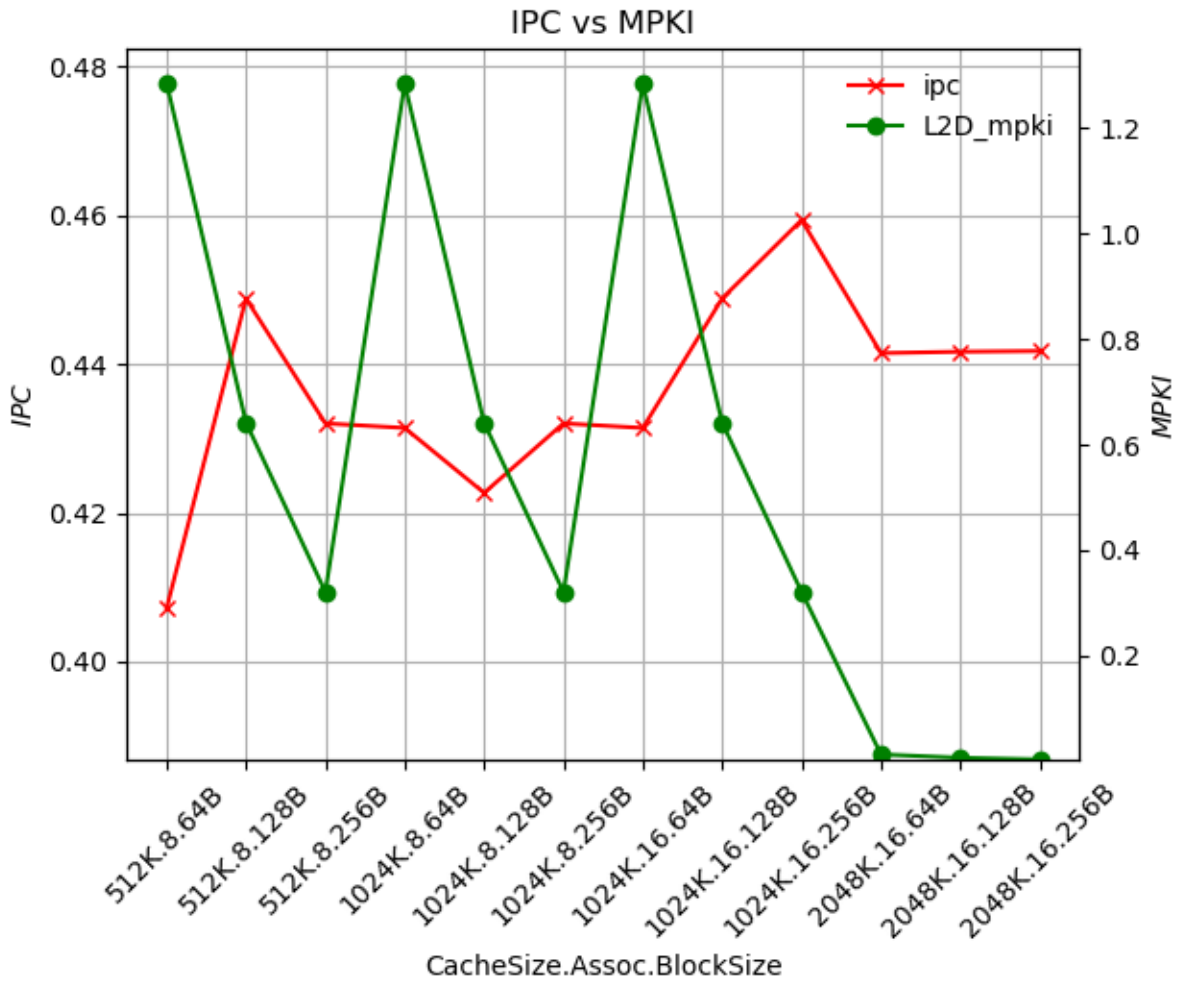
Σε αυτό το κομμάτι οι παράμετροι της L1 cache και του TLB διατηρούνται σταθερές και ίσες με:

- L1 size: 32 KB
- L1 associativity: 8
- L1 block size: 64 B
- TLB size (entries): 64
- TLB associativity: 4
- TLB page size: 4096 B

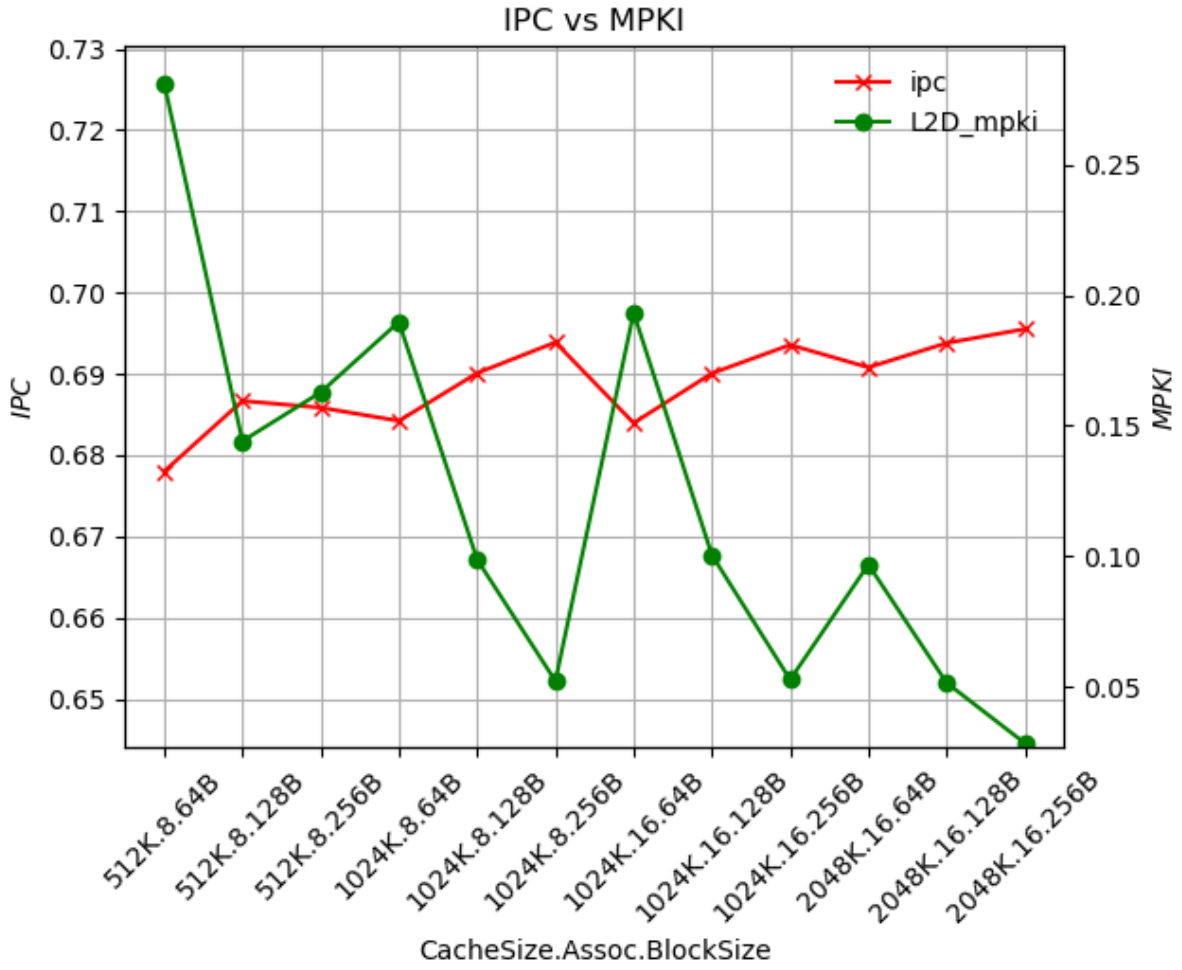
Όσον αφορά τα χαρακτηριστικά της L2 cache, έχουμε τις εξής περιπτώσεις:

L2 size (KB)	L2 associativity	L2 block size (B)
512	8	64
512	8	128
512	8	256
1024	8	64
1024	8	128
1024	8	256
1024	16	64
1024	16	128
1024	16	256
2048	16	64
2048	16	128
2048	16	256

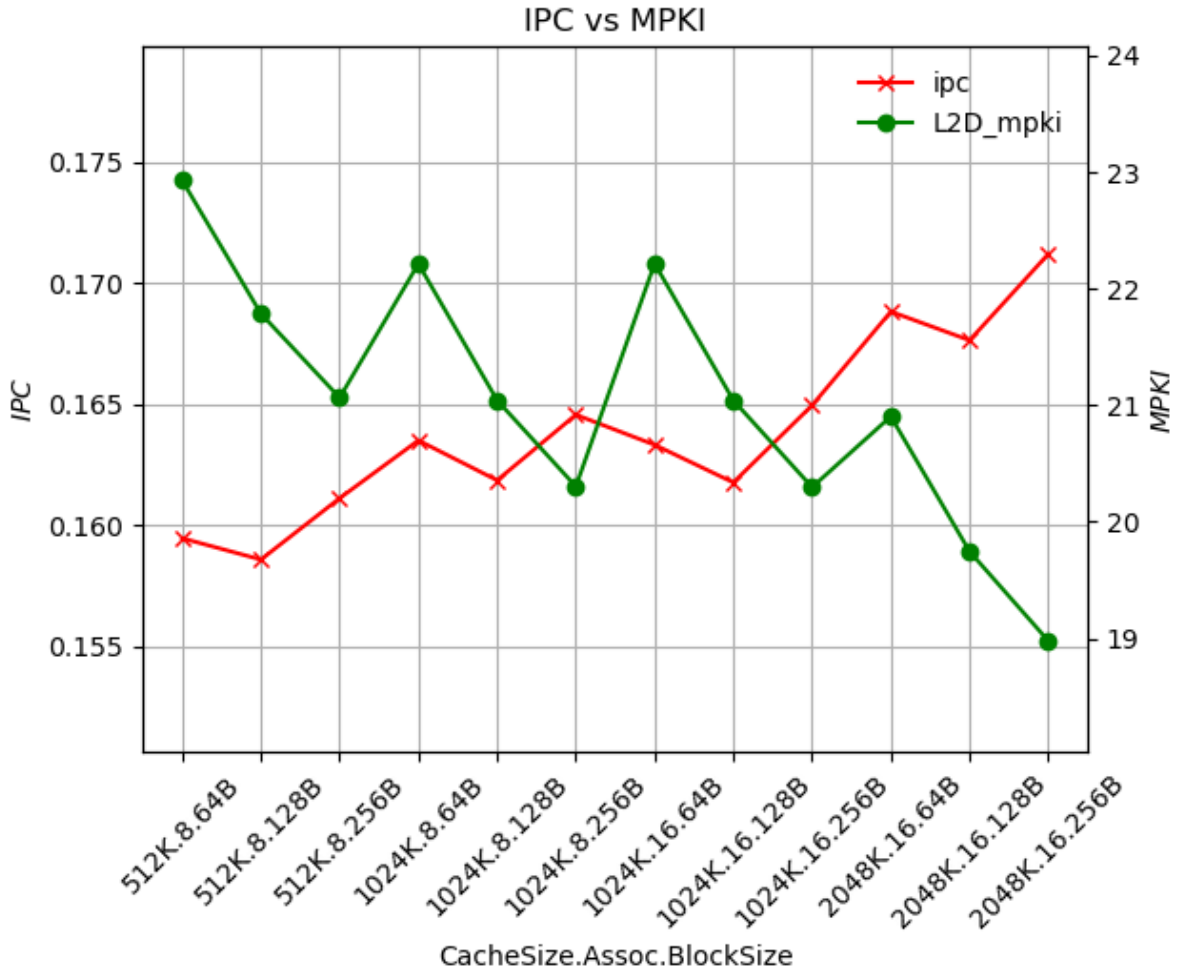
1.2.1 blackscholes



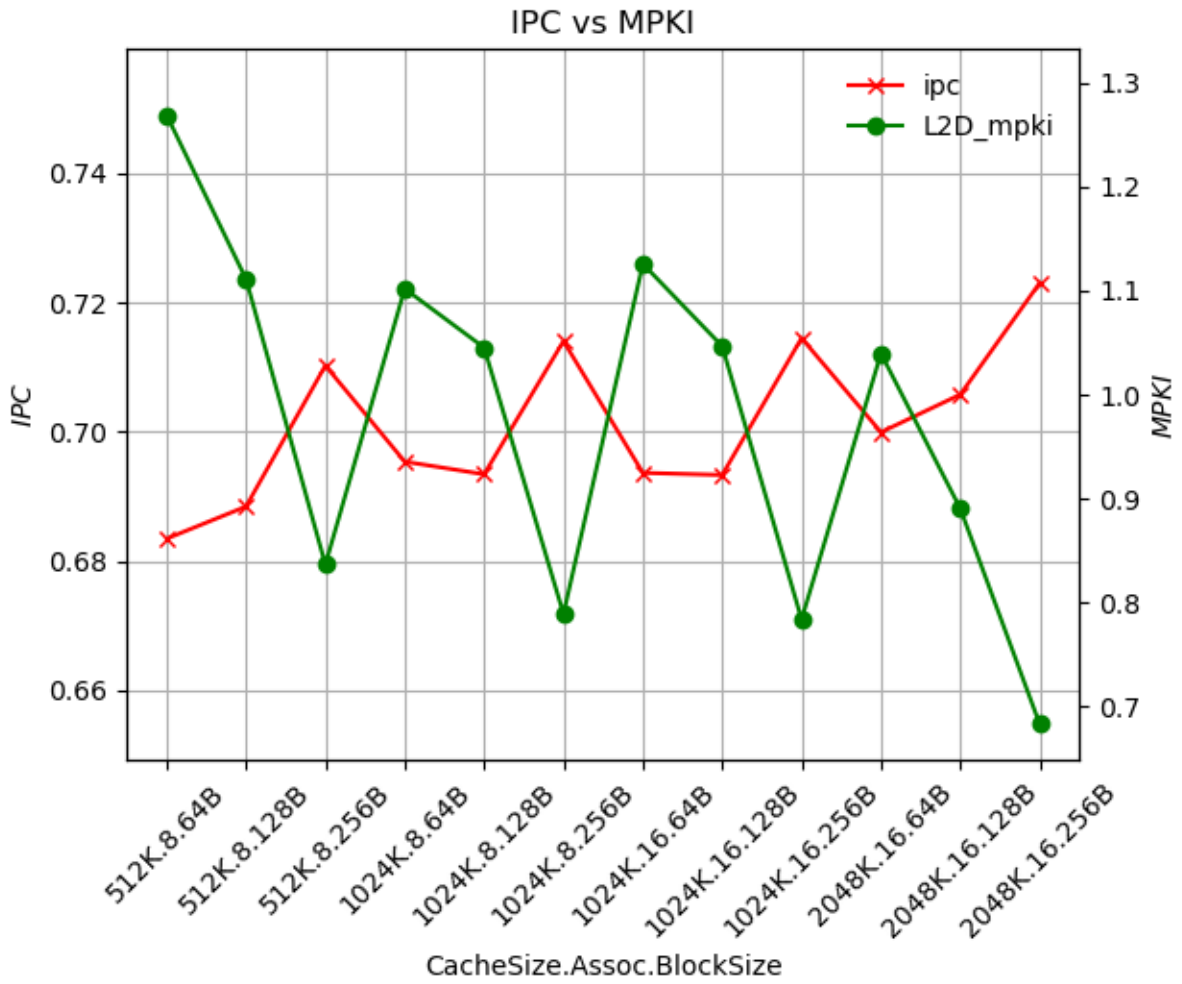
1.2.2 bodytrack



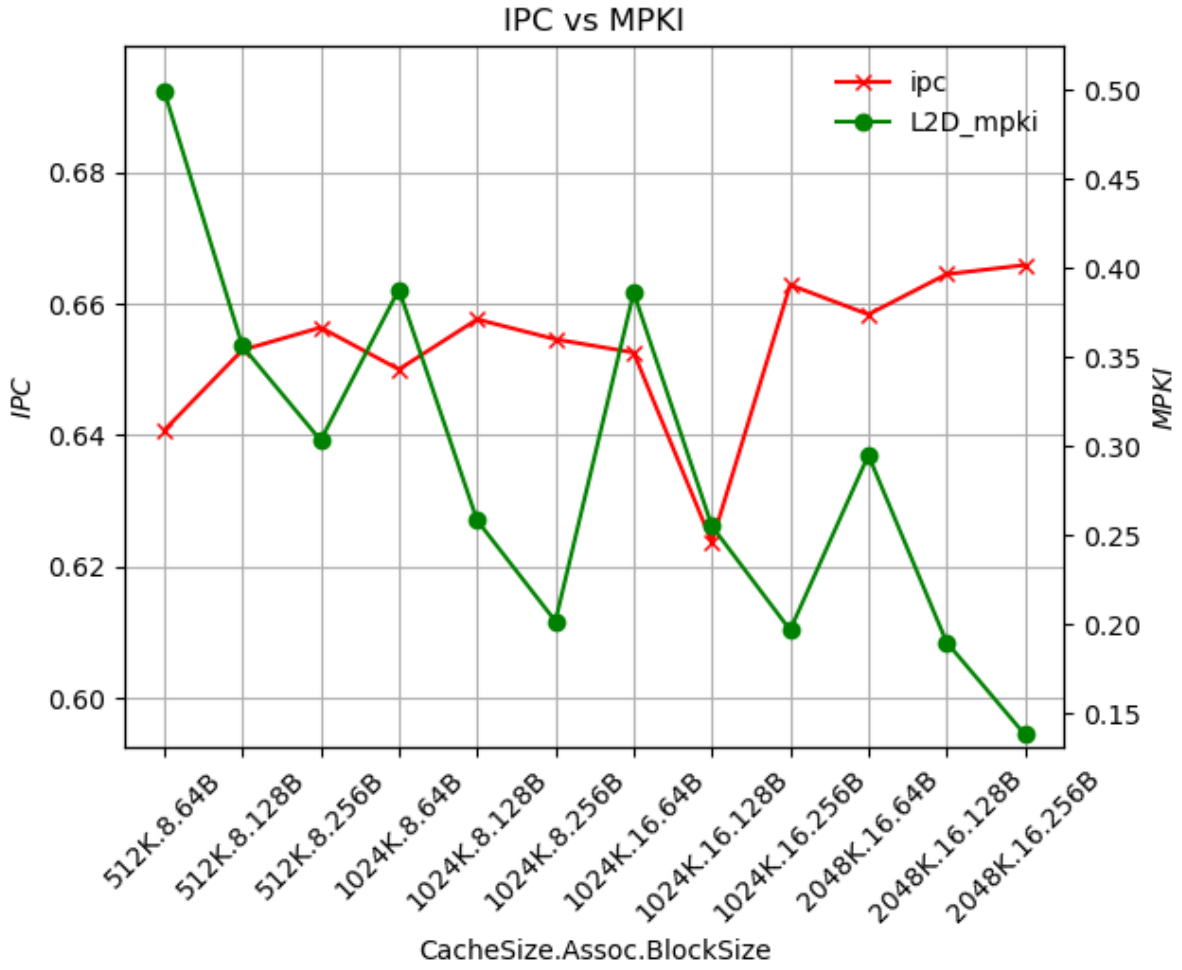
1.2.3 canneal


IPC vs MPKI
ipc
L2D_mpki
IPC
MPKI
0.175
0.170
0.165
0.160
0.155
24
23
22
21
20
19
512K.8.64B
512K.8.128B
512K.8.256B
1024K.8.64B
1024K.8.128B
1024K.8.256B
1024K.16.64B
1024K.16.128B
1024K.16.256B
2048K.16.64B
2048K.16.128B
2048K.16.256B
CacheSize.Assoc.BlockSize

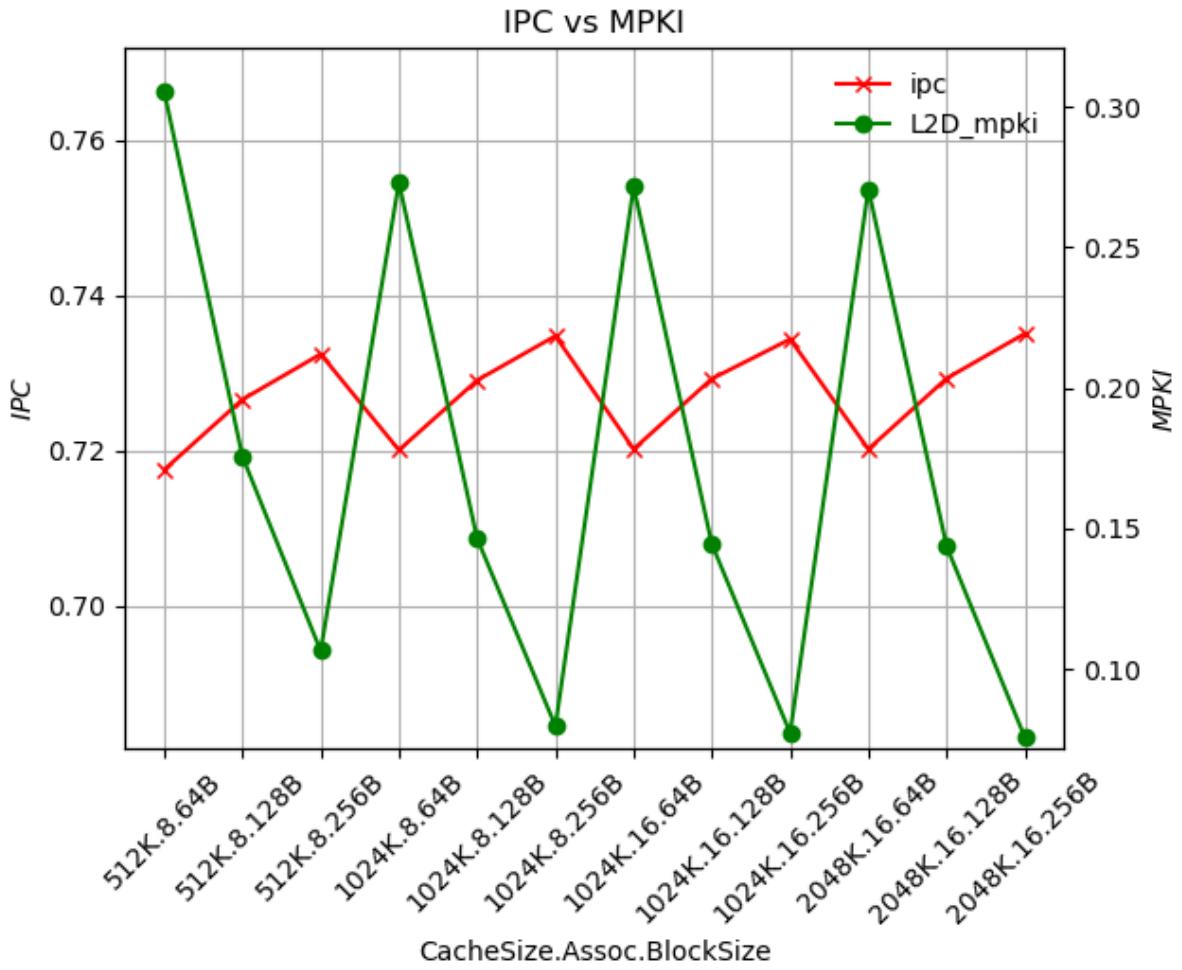
1.2.4 fluidanimate



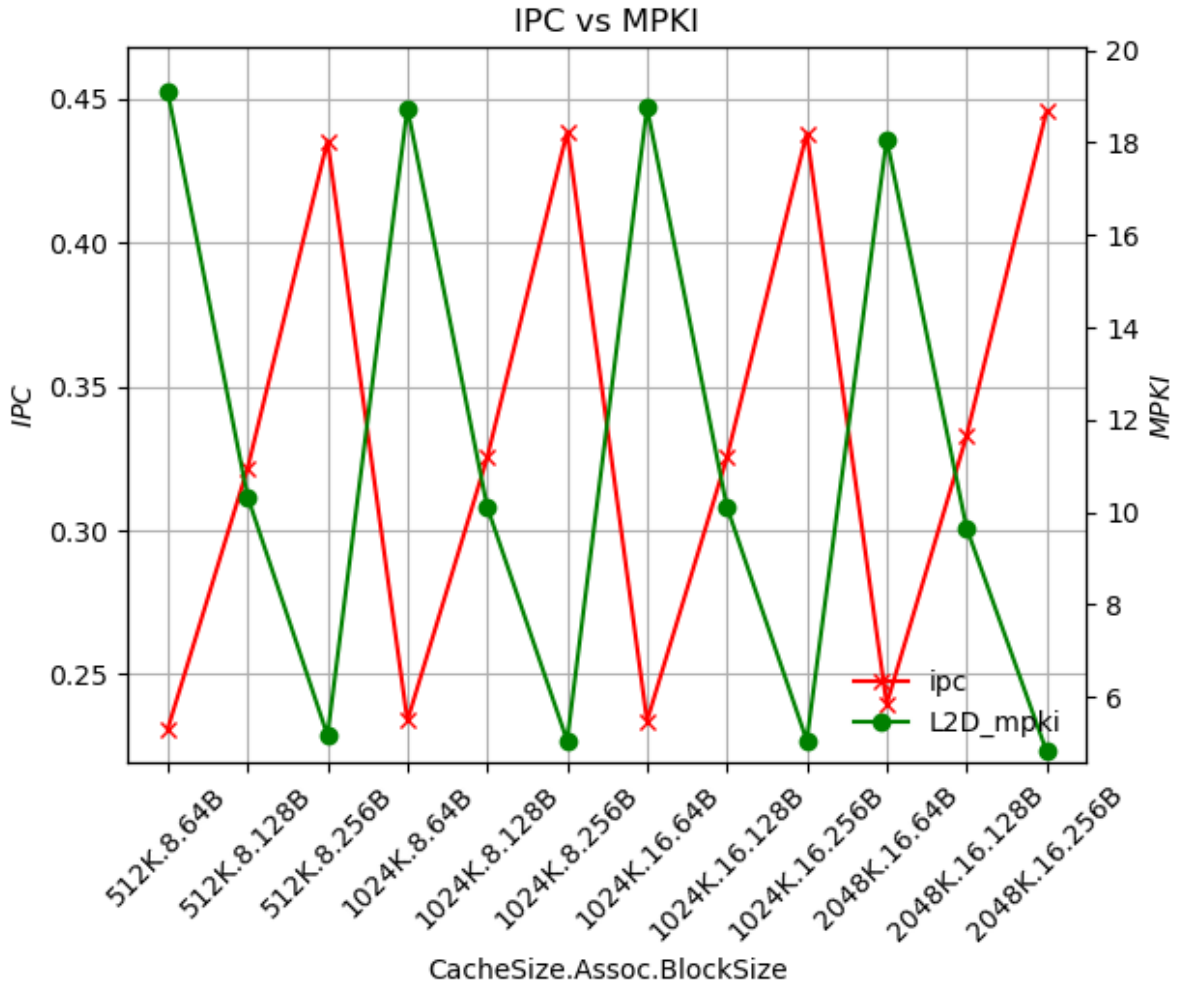
1.2.5 freqmine



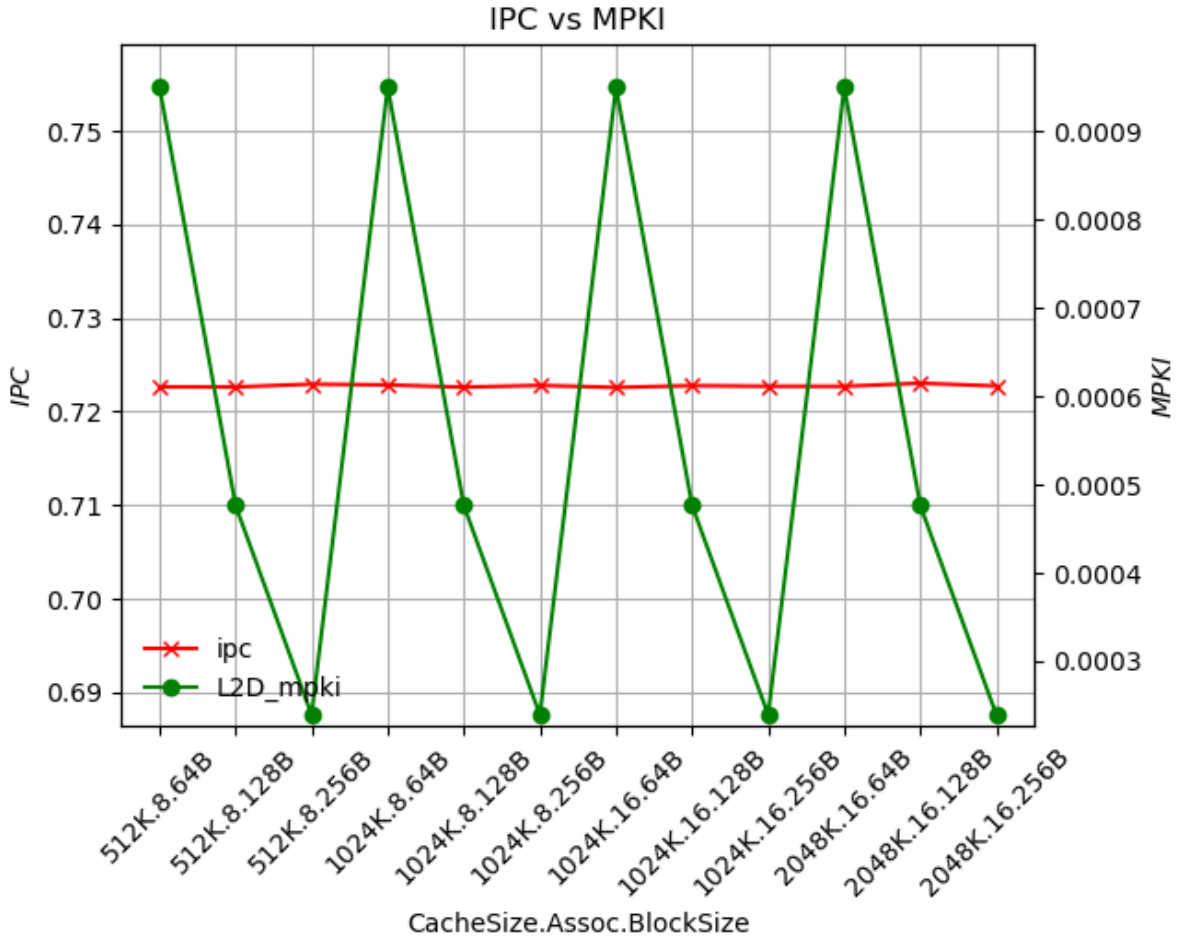
1.2.6 rtview


IPC vs MPKI
ipc
L2D_mpki
IPC
MPKI
0.76
0.74
0.72
0.70
0.30
0.25
0.20
0.15
0.10
512K.8.64B
512K.8.128B
512K.8.256B
1024K.8.64B
1024K.8.128B
1024K.8.256B
1024K.16.64B
1024K.16.128B
1024K.16.256B
2048K.16.64B
2048K.16.128B
2048K.16.256B
CacheSize.Assoc.BlockSize

1.2.7 streamcluster

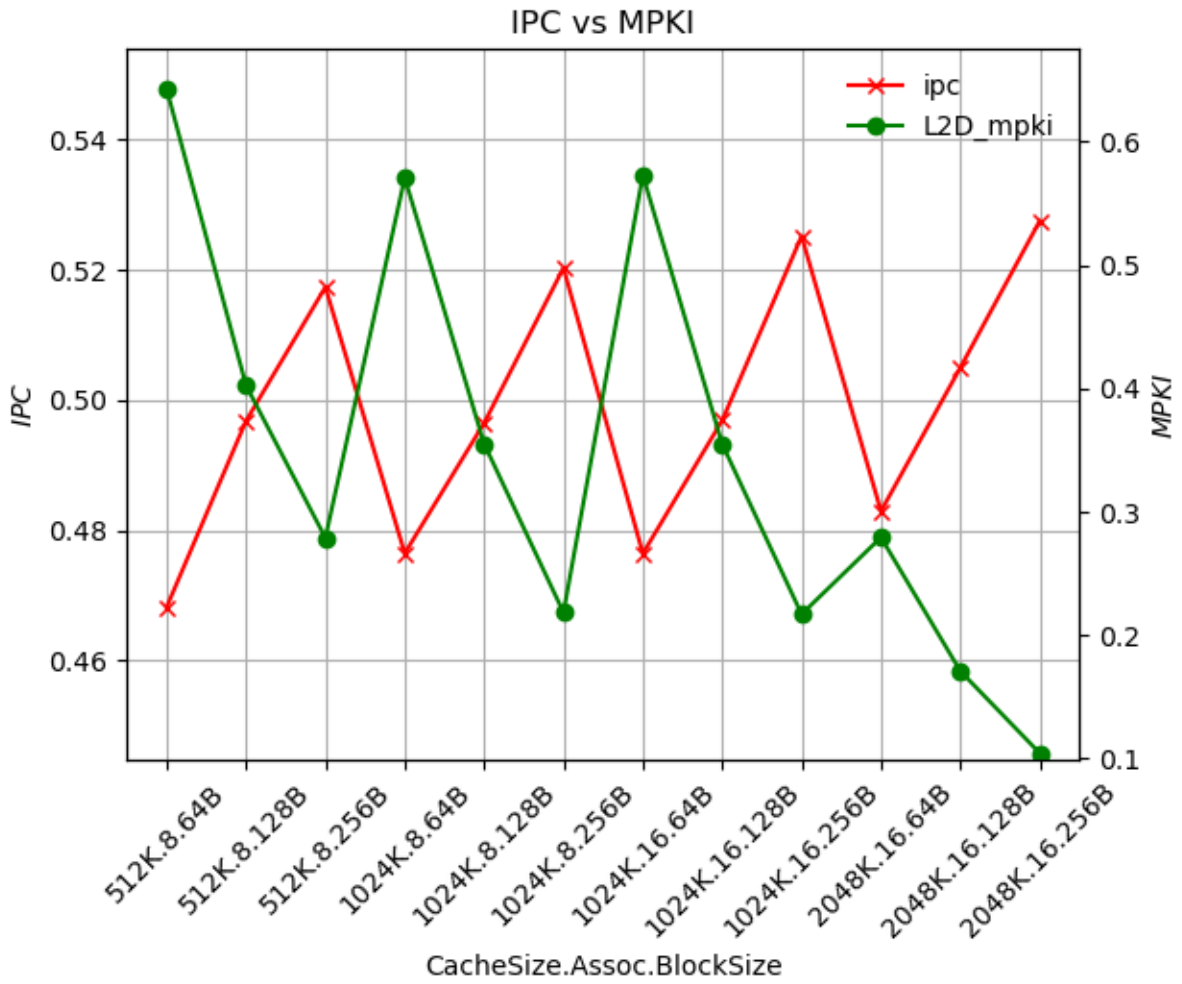


1.2.8 swaptions



Για τους ίδιους λόγους που εξηγήθηκαν παραπάνω, παρουσιάζουμε και τον γεωμετρικό μέσο των benchmarks:

1.2.9 Γεωμετρικός μέσος των benchmarks



1.2.10 Γενικές παρατηρήσεις για την L2

- Ό,τι αναφέρθηκε παραπάνω για την αντίστροφη σχέση IPC/MPKI εξακολουθεί να ισχύει.
- Καλύτερη επιλογή φαίνεται να είναι η τριπλέτα:

  `(cache size, associativity, block size) = (2048K, 16, 256B)`

- Η αύξηση του block size επιφέρει μείωση των compulsory misses και άρα αύξηση του IPC στα benchmarks bodytrack, fluidanimate, blackscholes, rtview, streamcluster.
- Στο blackscholes, φαίνεται ότι για cache size 1024K έχουμε σημαντική βελτίωση της επίδοσης, η οποία δεν μπορεί να βελτιωθεί παραπάνω με αύξηση των υπολοίπων παραμέτρων.
- Στα benchmarks canneal, fluidanimate παρατηρούμε ότι όταν αυξάνουμε το μέγεθος της cache μειώνονται τα capacity misses και άρα βελτιώνεται το IPC.
- Το swaptions φαίνεται να μην επηρεάζεται σχεδόν καθόλου από τις αλλαγές στα χαρακτηριστικά (IPC σχεδόν σταθερό), οπότε μπορεί να χρησιμοποιούνται συνεχώς τα ίδια blocks, ή να μην χρειάζονται ξανά όσα φεύγουν από την cache.

- Γενικά, το block size παίζει αρκετά σημαντικό ρόλο εδώ, αν και οι άλλες δύο παράμετροι έχουν κι αυτές κάποια σημασία.

1.3 TLB

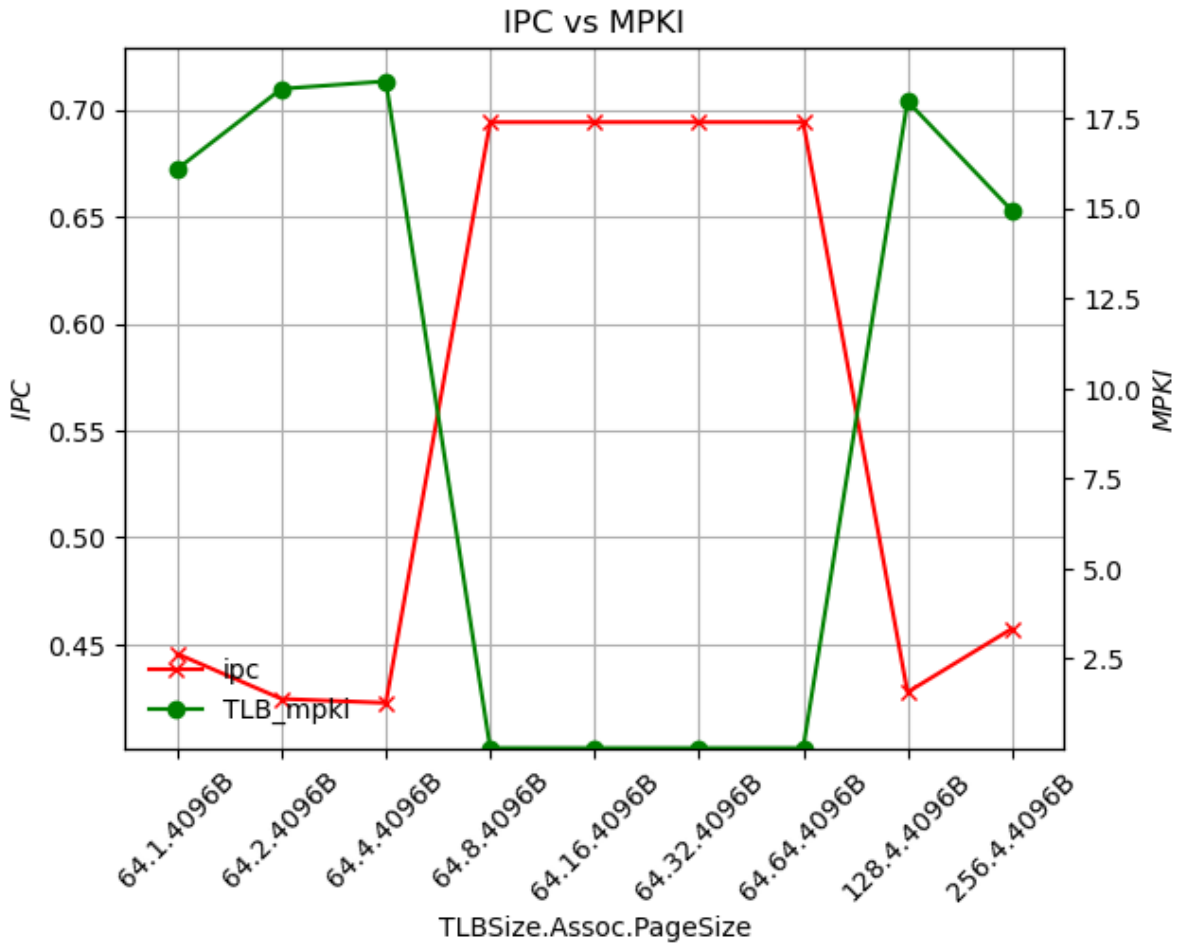
Σε αυτό το κομμάτι οι παράμετροι της L1 cache και της L2 cache διατηρούνται σταθερές και ίσες με:

- L1 size: 32 KB
- L1 associativity: 8
- L1 block size: 64 B
- L2 size: 1024 KB
- L2 associativity: 8
- L2 block size: 128 B

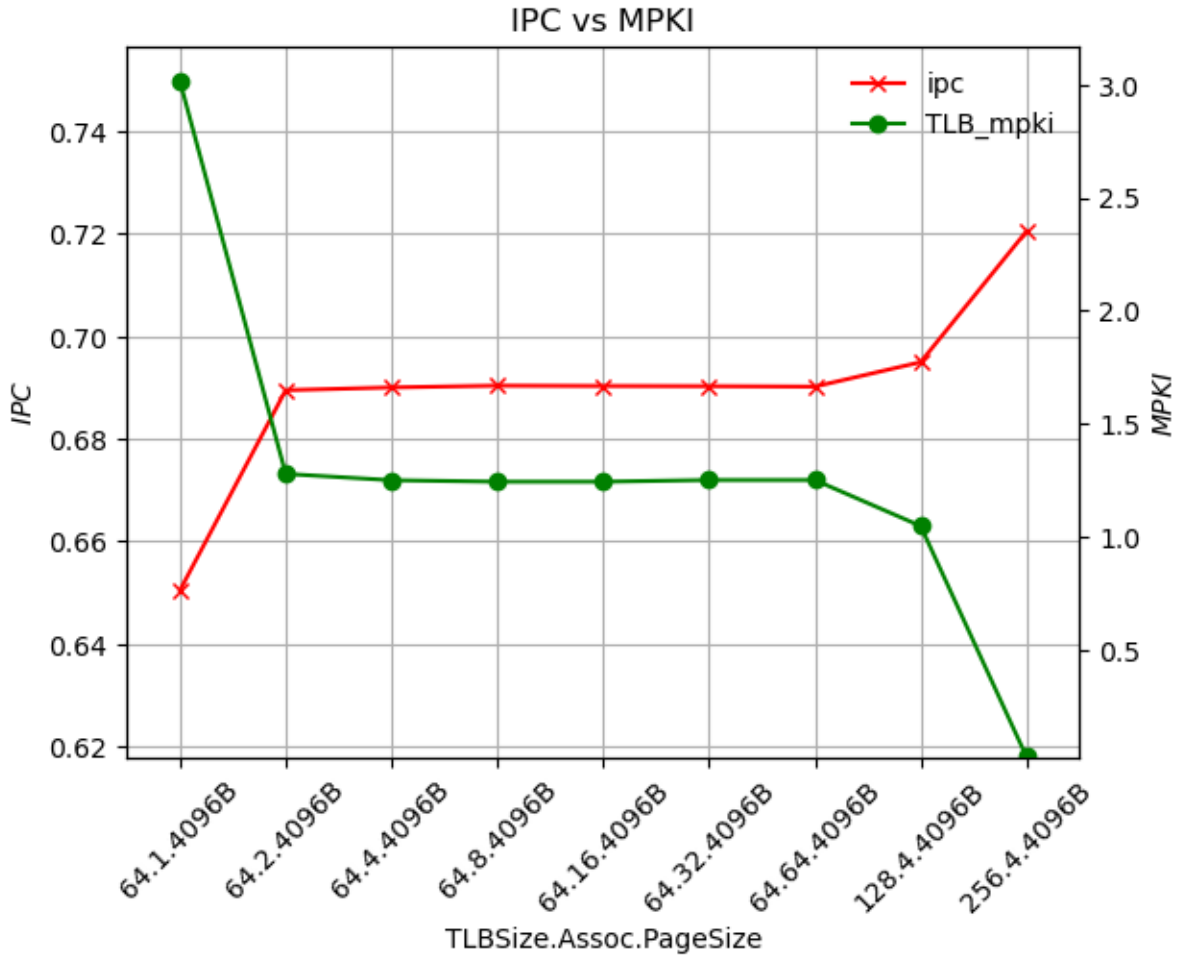
Όσον αφορά τα χαρακτηριστικά του TLB, έχουμε τις εξής περιπτώσεις:

TLB size (entries)	TLB associativity	TLB page size (B)
64	1	4096
64	2	4096
64	4	4096
64	8	4096
64	16	4096
64	32	4096
64	64	4096
128	4	4096
256	4	4096

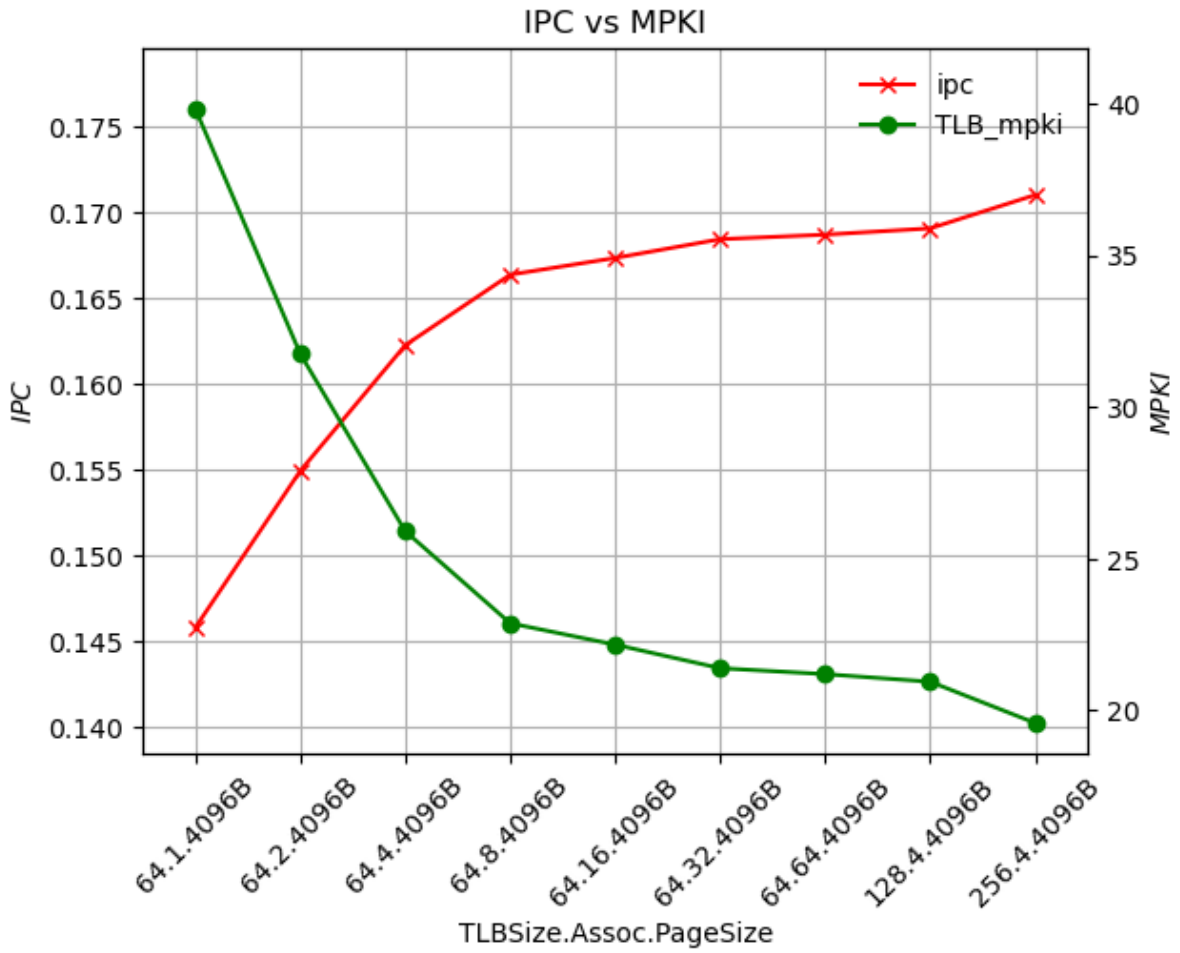
1.3.1 blackscholes



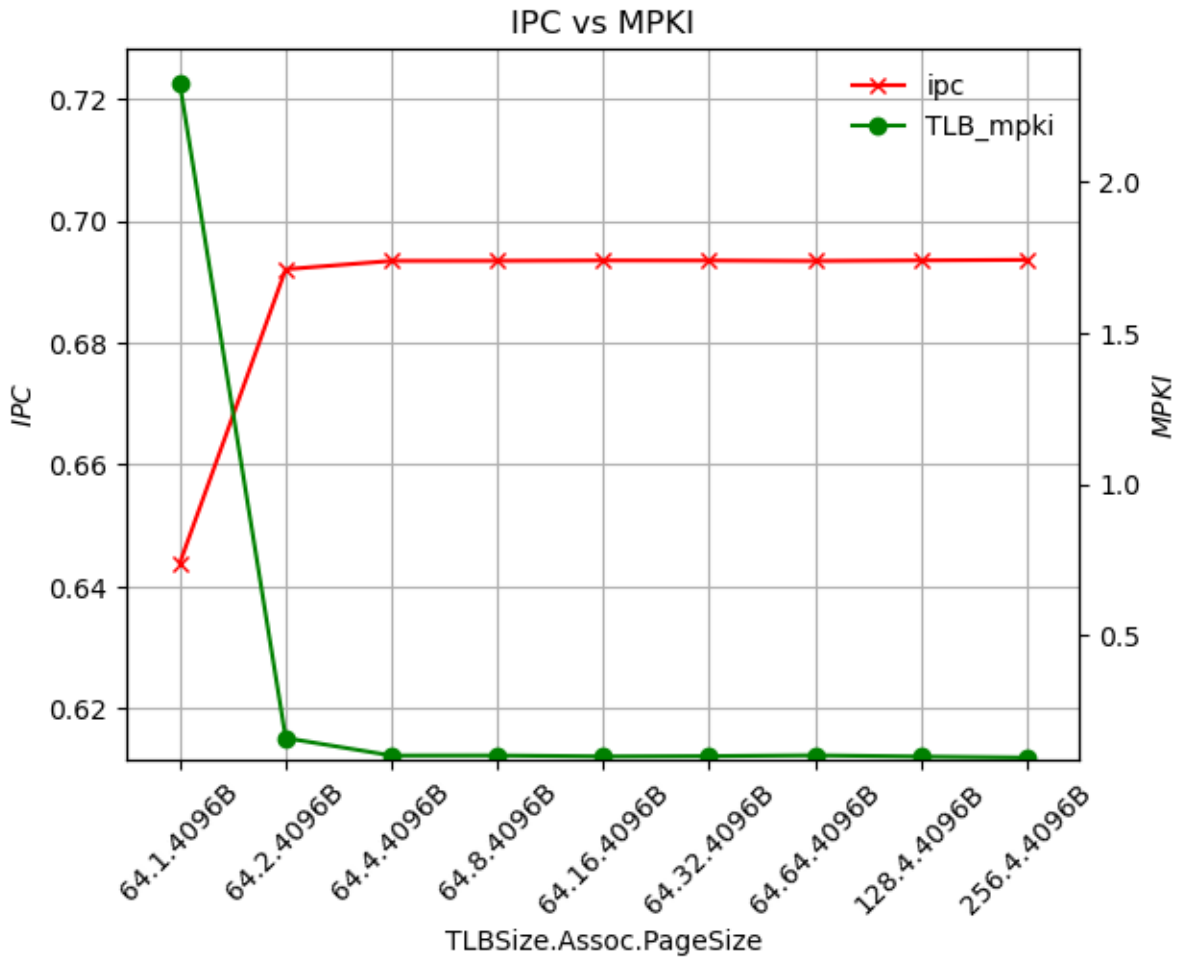
1.3.2 bodytrack



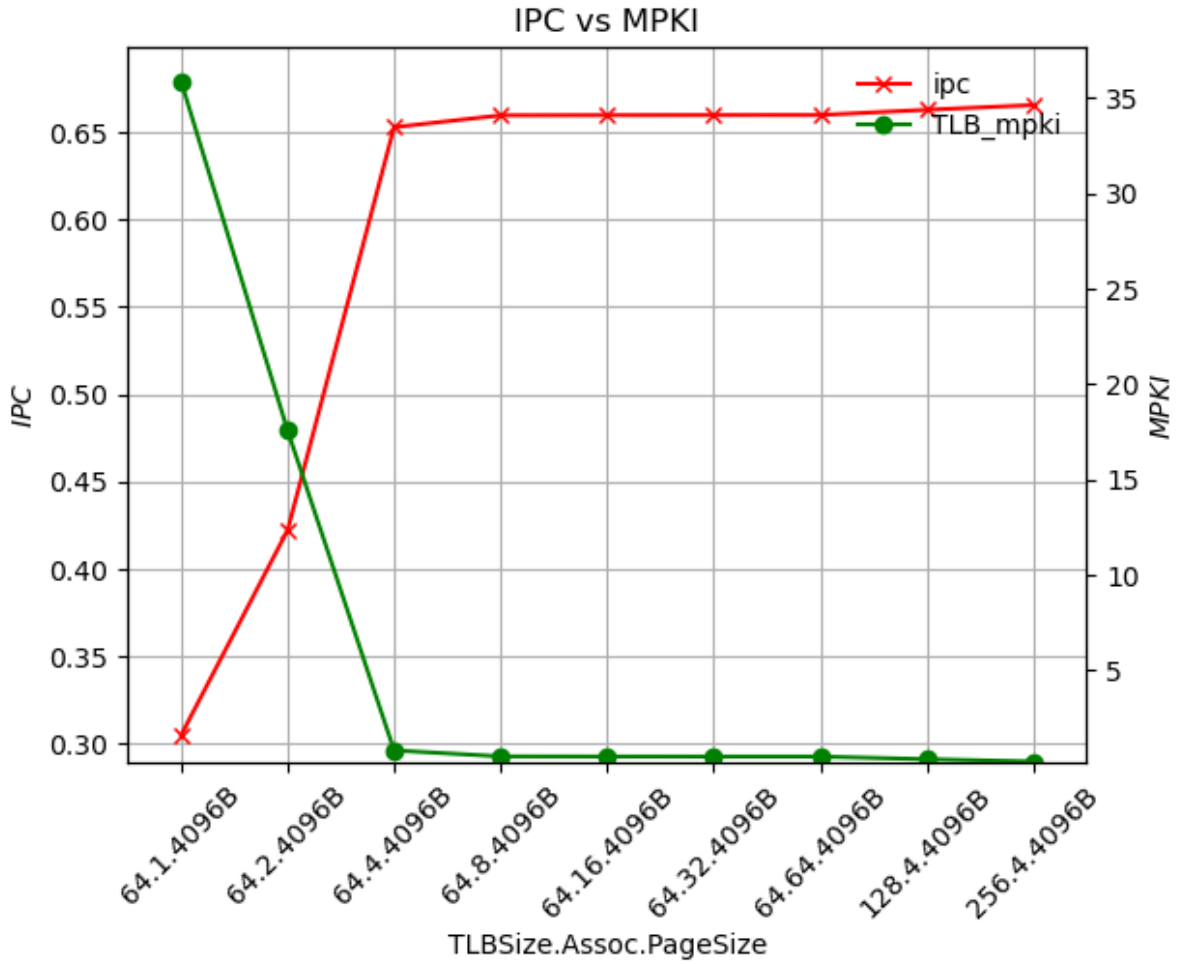
1.3.3 canneal


IPC vs MPKI
ipc
TLB_mpki
IPC
MPKI
0.175
0.170
0.165
0.160
0.155
0.150
0.145
0.140
40
35
30
25
20
64.1.4096B
64.2.4096B
64.4.4096B
64.8.4096B
64.16.4096B
64.32.4096B
64.64.4096B
128.4.4096B
256.4.4096B
TLBSize.Assoc.PageSize

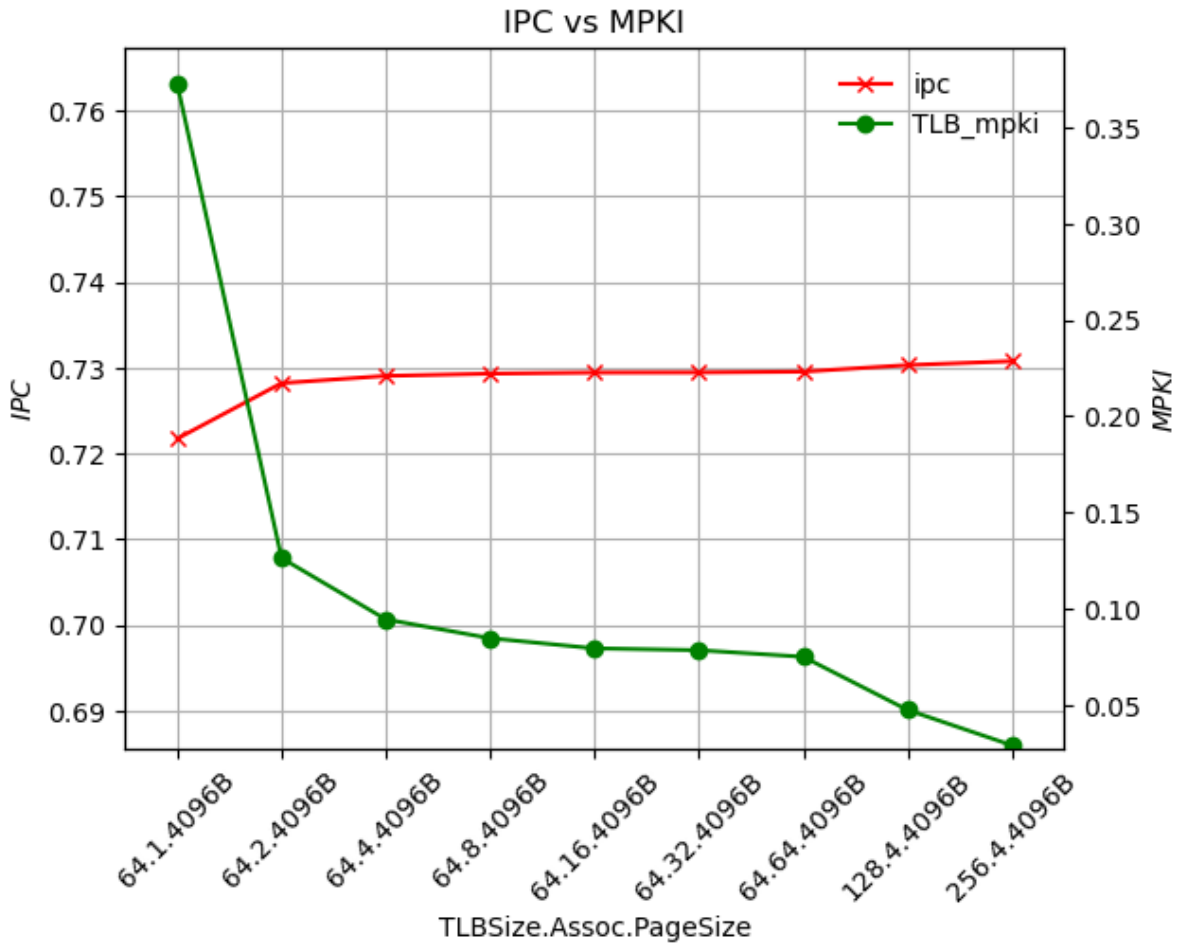
1.3.4 fluidanimate



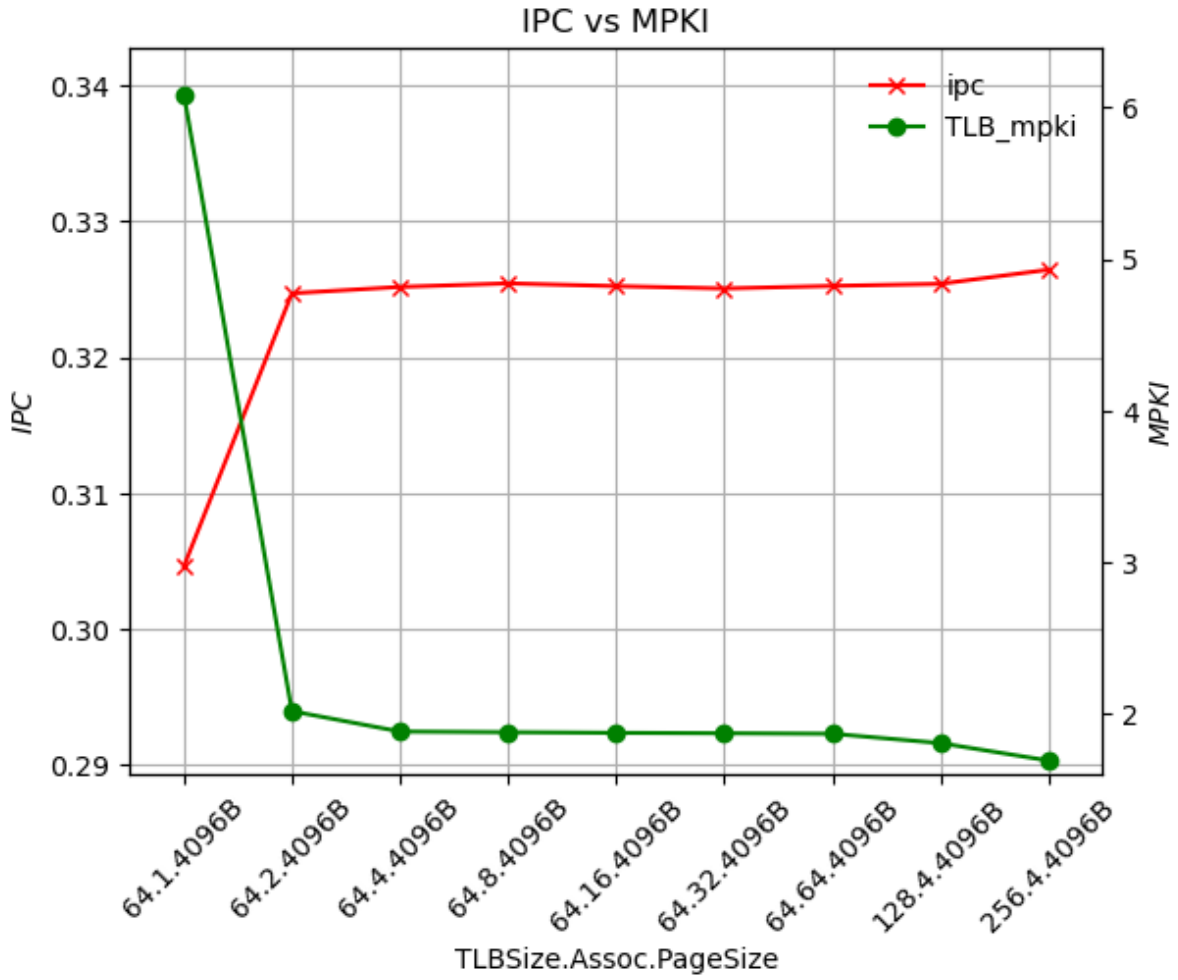
1.3.5 freqmine



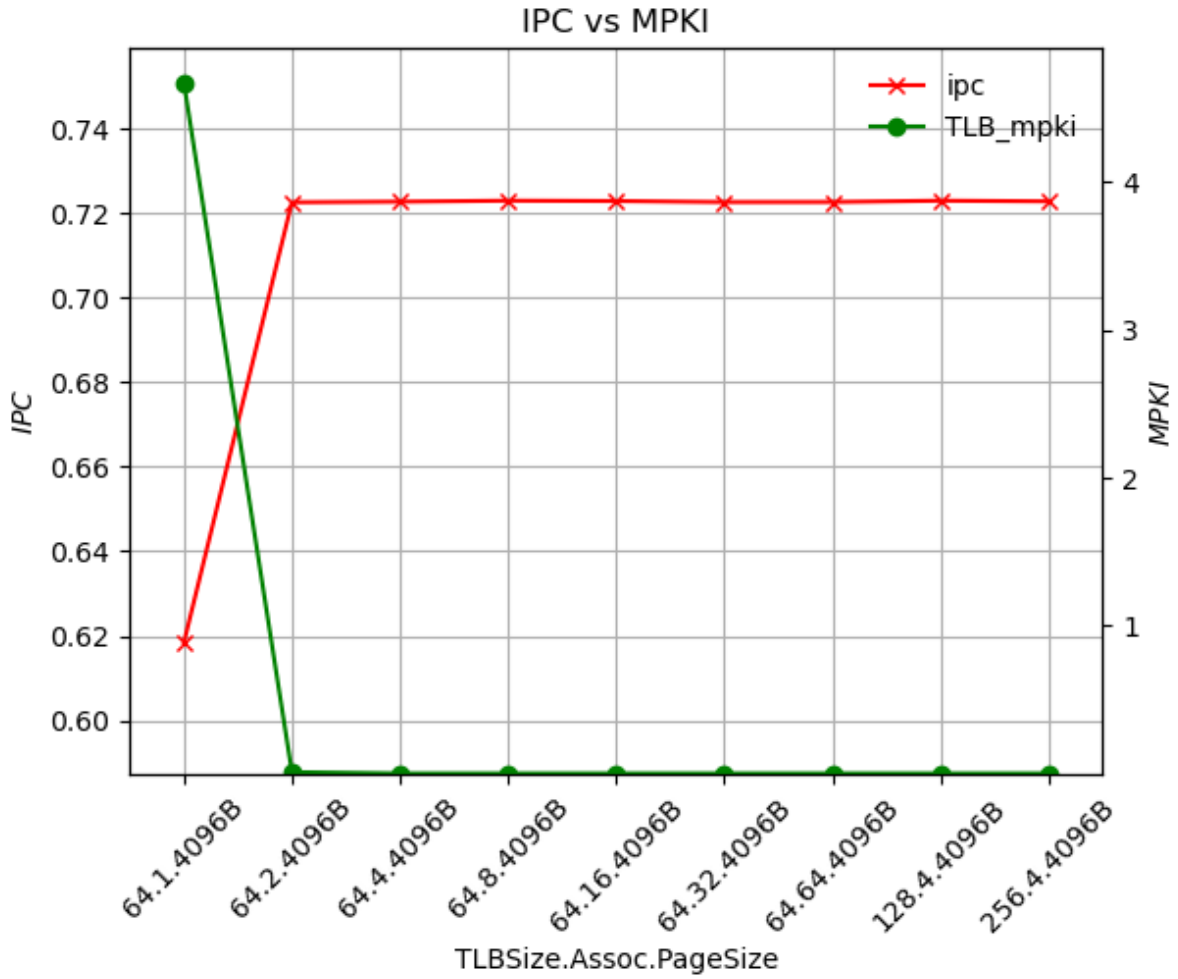
1.3.6 rtview


IPC vs MPKI
ipc
TLB_mpki
IPC
MPKI
0.76
0.75
0.74
0.73
0.72
0.71
0.70
0.69
0.35
0.30
0.25
0.20
0.15
0.10
0.05
64.1.4096B
64.2.4096B
64.4.4096B
64.8.4096B
64.16.4096B
64.32.4096B
64.64.4096B
128.4.4096B
256.4.4096B
TLBSize.Assoc.PageSize

1.3.7 streamcluster

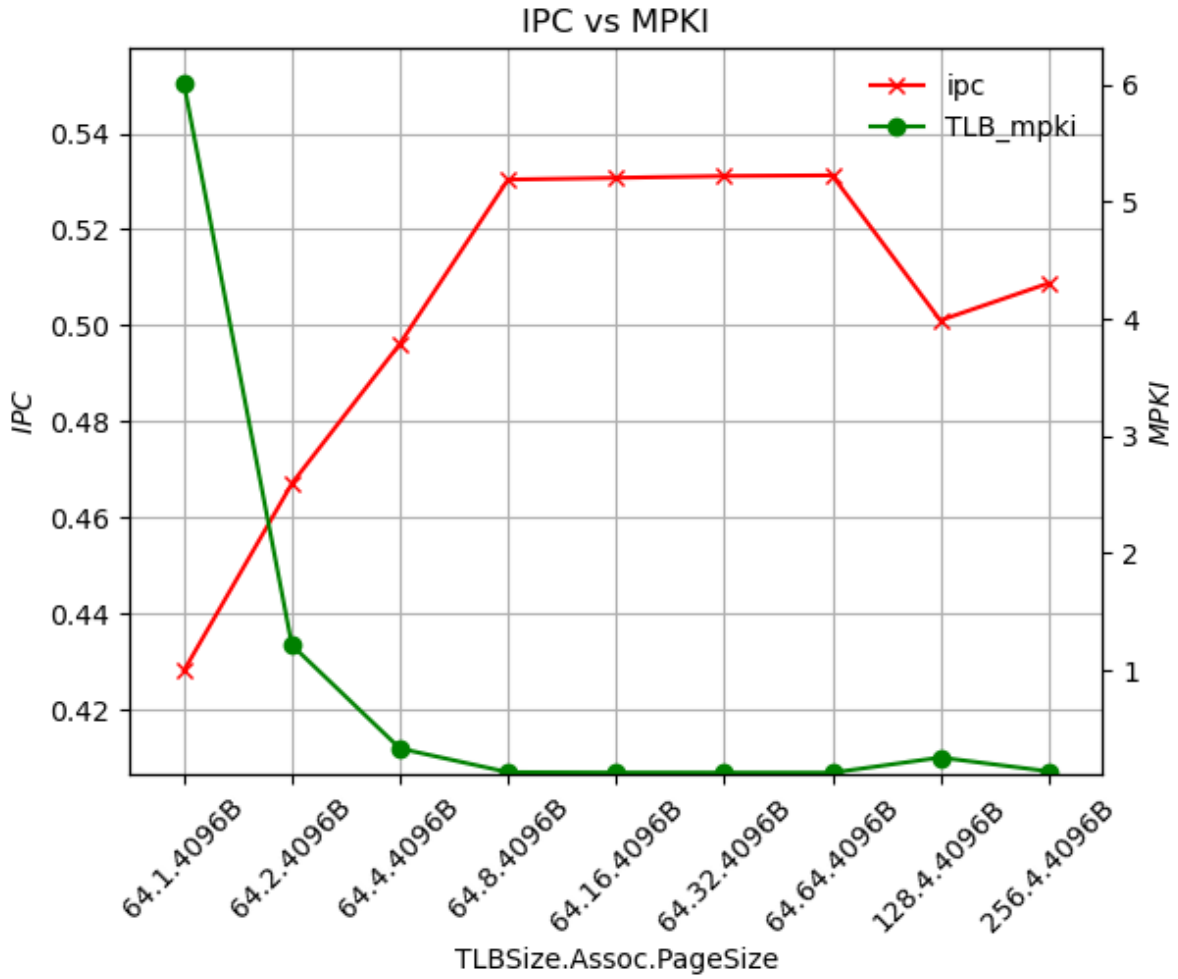


1.3.8 swaptions



Και εδώ δείχνουμε τον γεωμετρικό μέσο:

1.3.9 Γεωμετρικός μέσος των benchmarks



1.3.10 Γενικές παρατηρήσεις για το TLB

- IPC και MPKI εξακολουθού να είναι αντιστρόφως ανάλογα.
- Καλύτερη επιλογή φαίνεται να είναι η τριπλέτα:

  `(TLB entries, associativity, page size) = (64, 8-64, 4096B)`

  όπου το associativity δεν παίζει σχεδόν καθόλου ρόλο στην επιλογή (λογικό είναι λοιπόν να επιλέξουμε το 8, αν μεγαλύτερο associativity σημαίνει μεγαλύτερο κόστος κατασκευής).
- Όλα τα benchmarks αυτή τη φορά επηρεάστηκαν με κάποιο τρόπο από την μεταβολή των χαρακτηριστικών (σε αντίθεση με τις περιπτώσεις L1 και L2 που εξετάστηκαν παραπάνω, όπου τα rtview και swaptions αντίστοιχα έμεναν σχετικά σταθερά)
- Το page size δεν μπορεί να αξιολογηθεί από τα παραπάνω, αφού είναι σε όλες τις περιπτώσεις ίσο με 4096B.
- Η αύξηση του TLB size επιδρά πολύ λίγο στην βελτίωση του IPC.

- Η αύξηση του associativity φαίνεται να επιδρά πολύ θετικά στο IPC, ωστόσο κάποια στιγμή επέρχεται ένας κορεσμός (περίπου στο associativity = 8 και μετά).

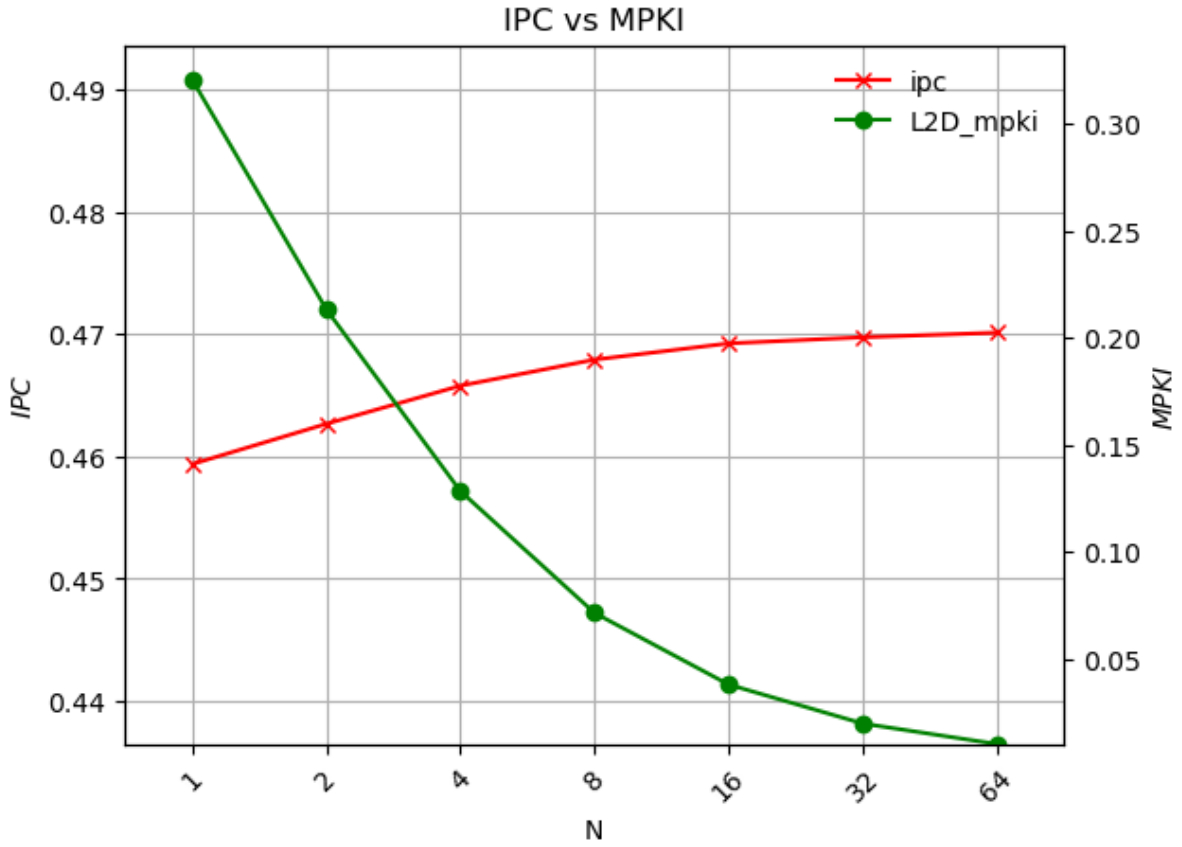
1.4 Prefetching

Σε αυτό το κομμάτι οι παράμετροι των L1 cache, L2 cache και TLB διατηρούνται σταθερές και ίσες με:

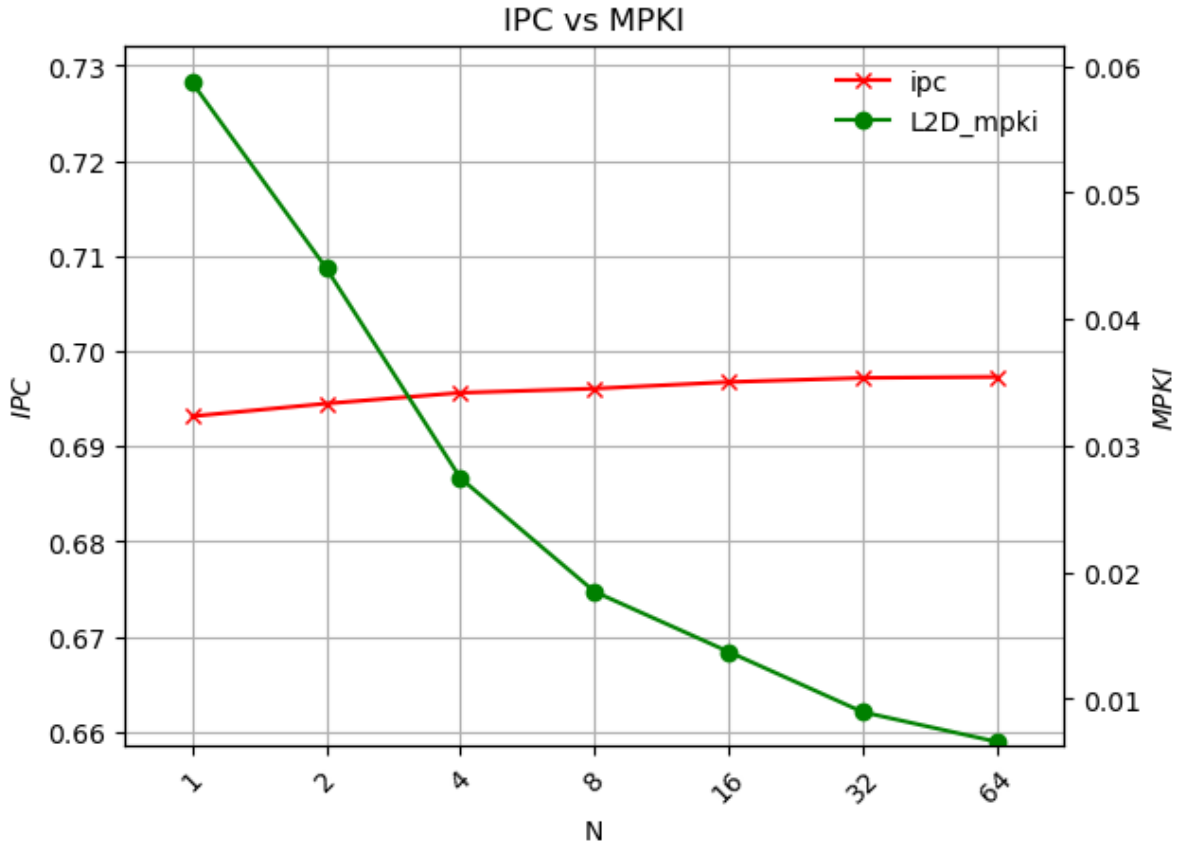
- L1 size: 32 KB
- L1 associativity: 8
- L1 block size: 64 B
- L2 size: 1024 KB
- L2 associativity: 8
- L2 block size: 128 B
- TLB size (entries): 64
- TLB associativity: 4
- TLB page size: 4096 B

Σε αντίθεση με τα παραπάνω, όπου το prefetching ήταν απενεργοποιημένο, θα έχουμε NEXT-N-LINE prefetching στην L2. Το N θα παίρνει τιμές {1, 2, 4, 8, 16, 32, 64}.

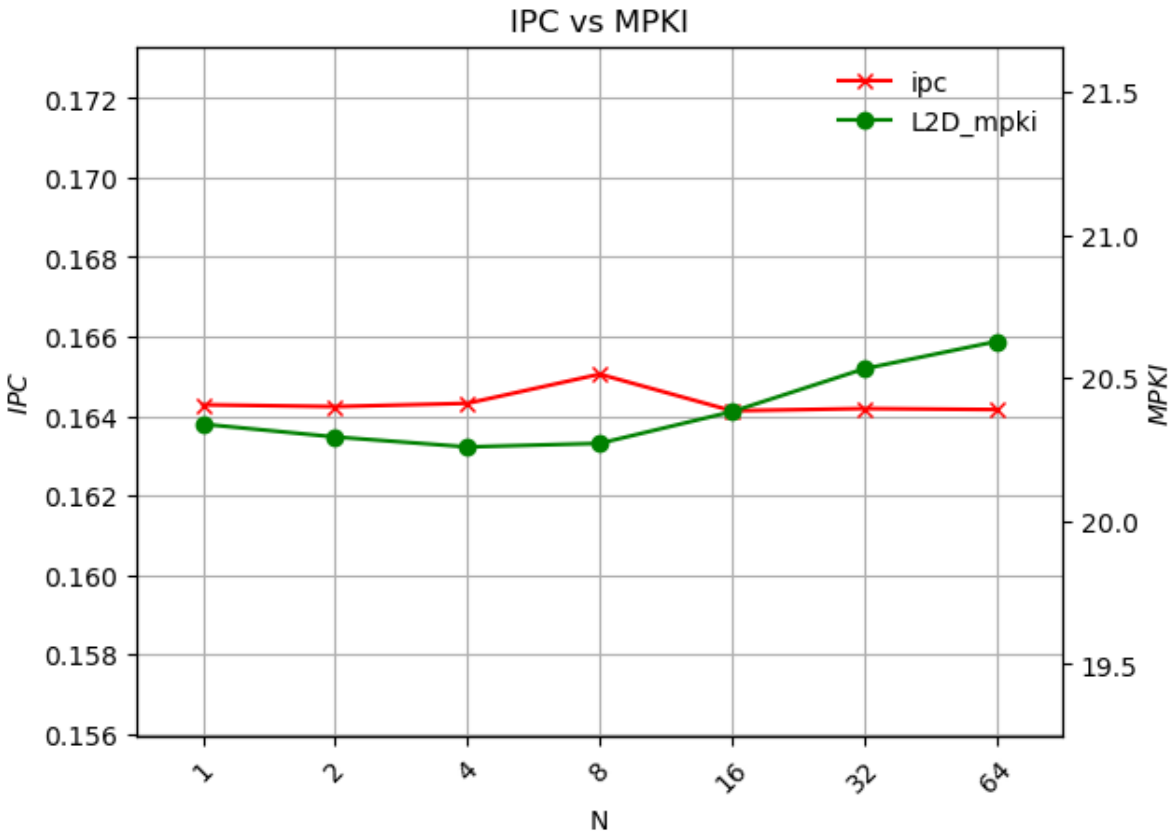
1.4.1 blackscholes



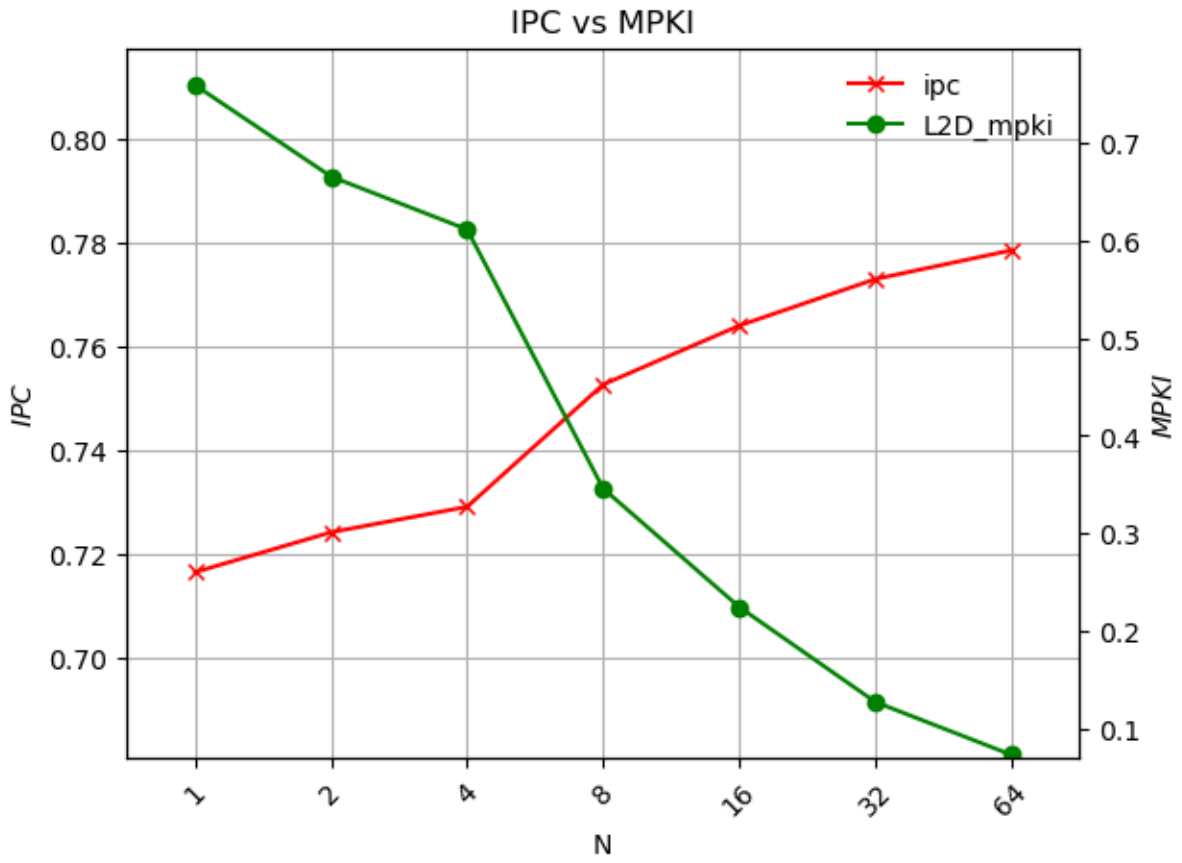
1.4.2 bodytrack



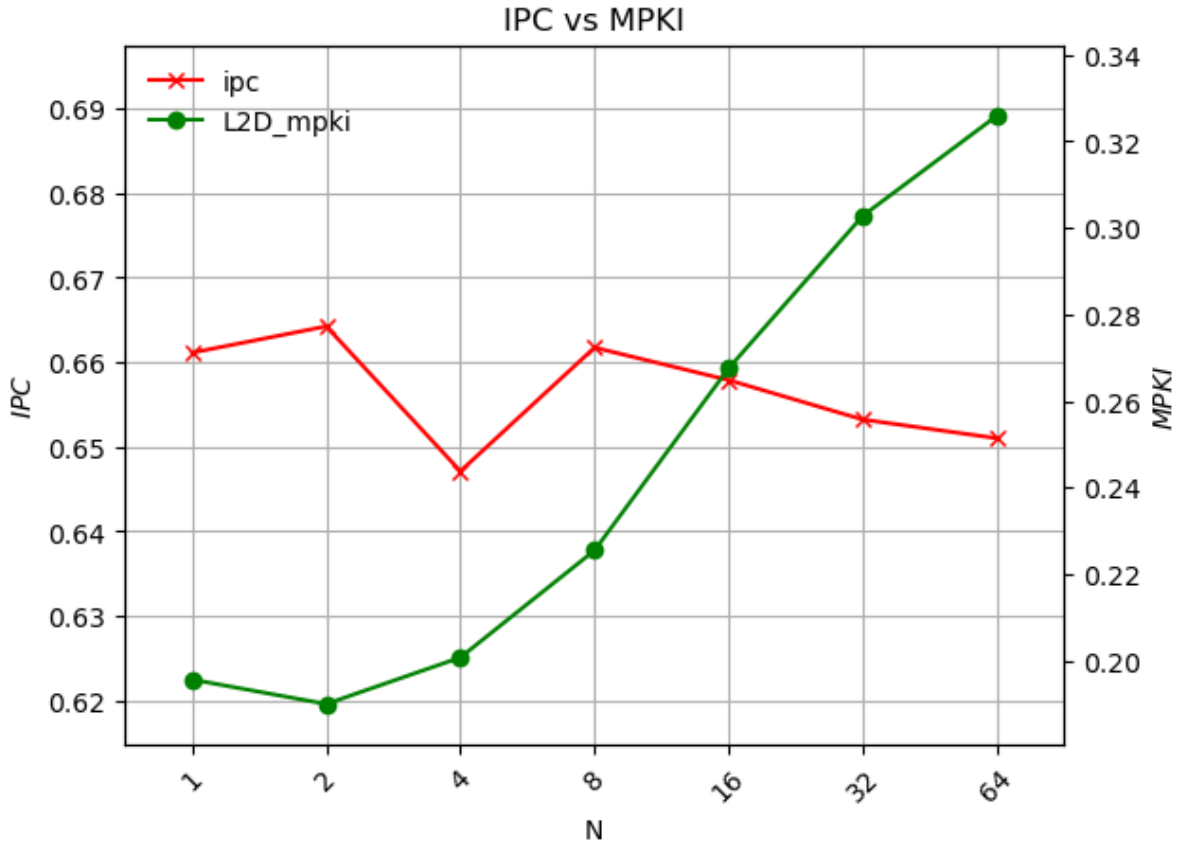
1.4.3 canneal


IPC vs MPKI
ipc
L2D_mpki
IPC
MPKI
N
0.172
0.170
0.168
0.166
0.164
0.162
0.160
0.158
0.156
21.5
21.0
20.5
20.0
19.5
1
2
4
8
16
32
64

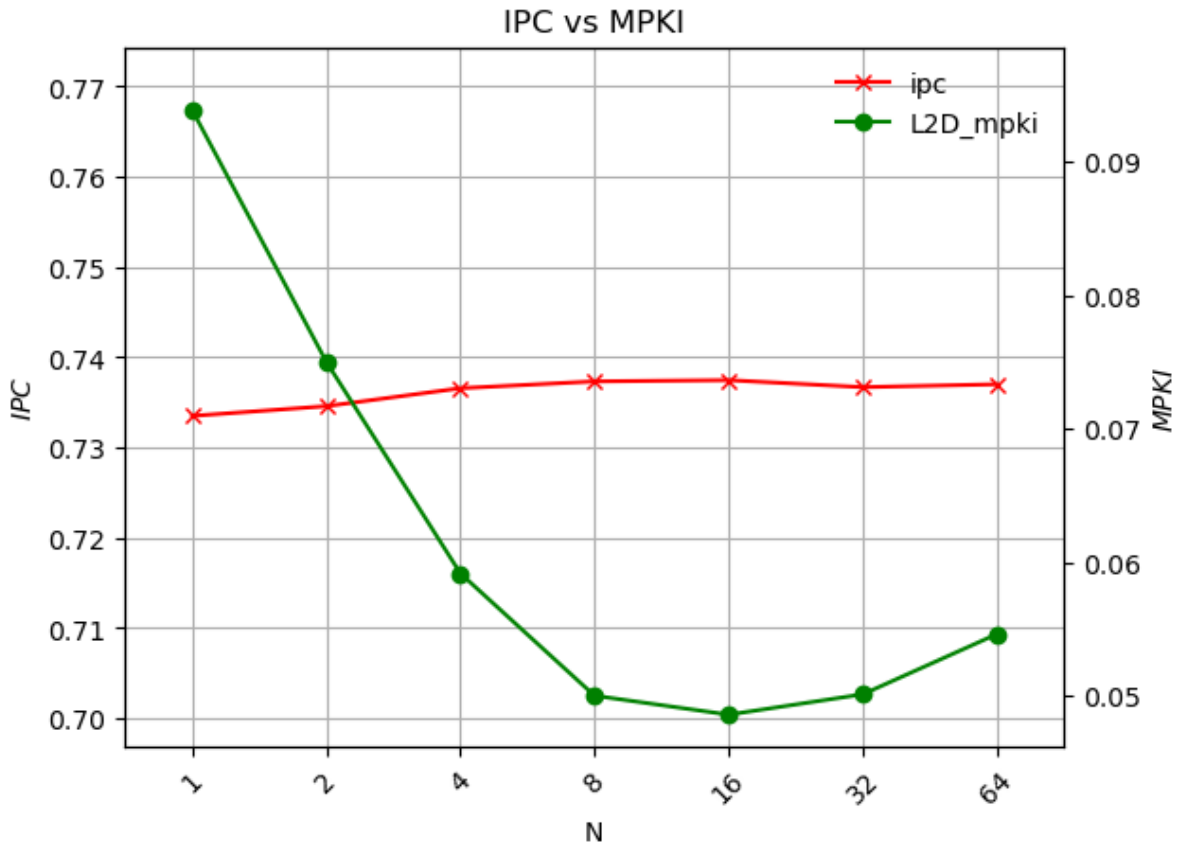
1.4.4 fluidanimate



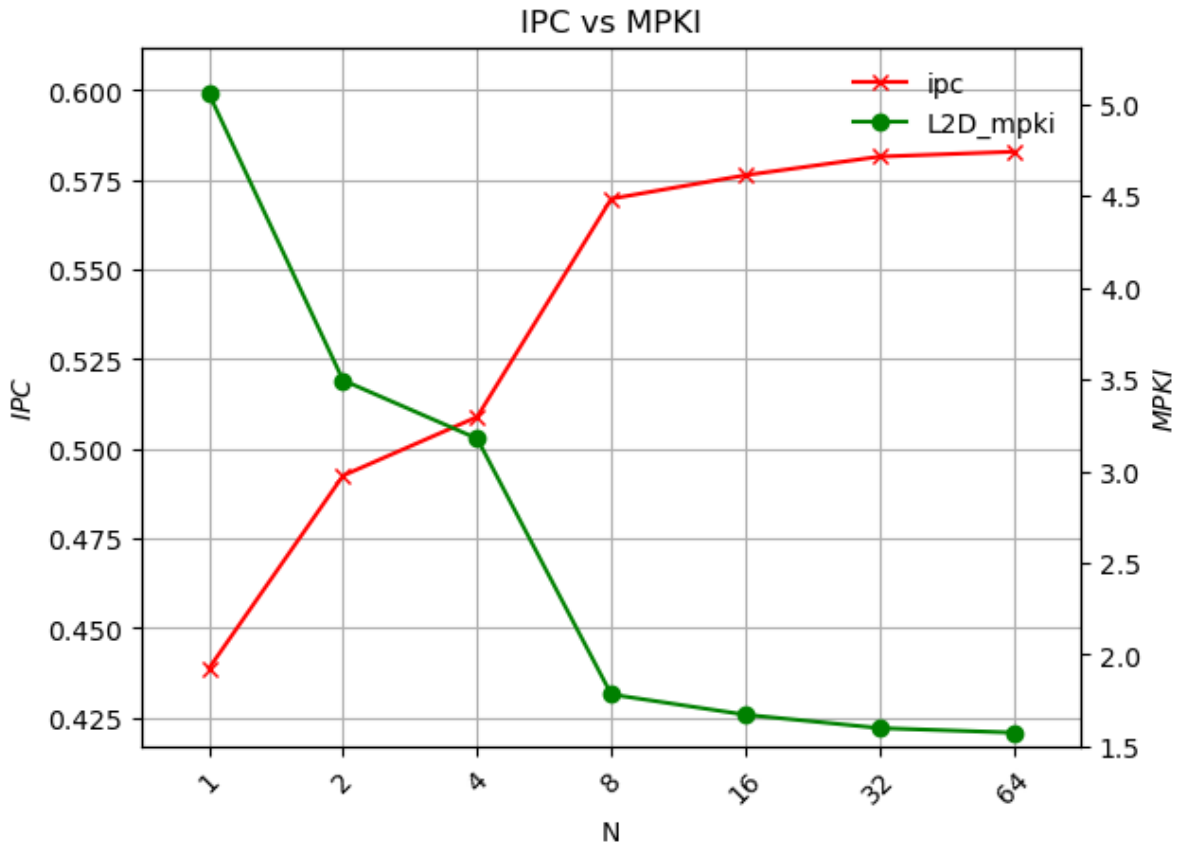
1.4.5 freqmine



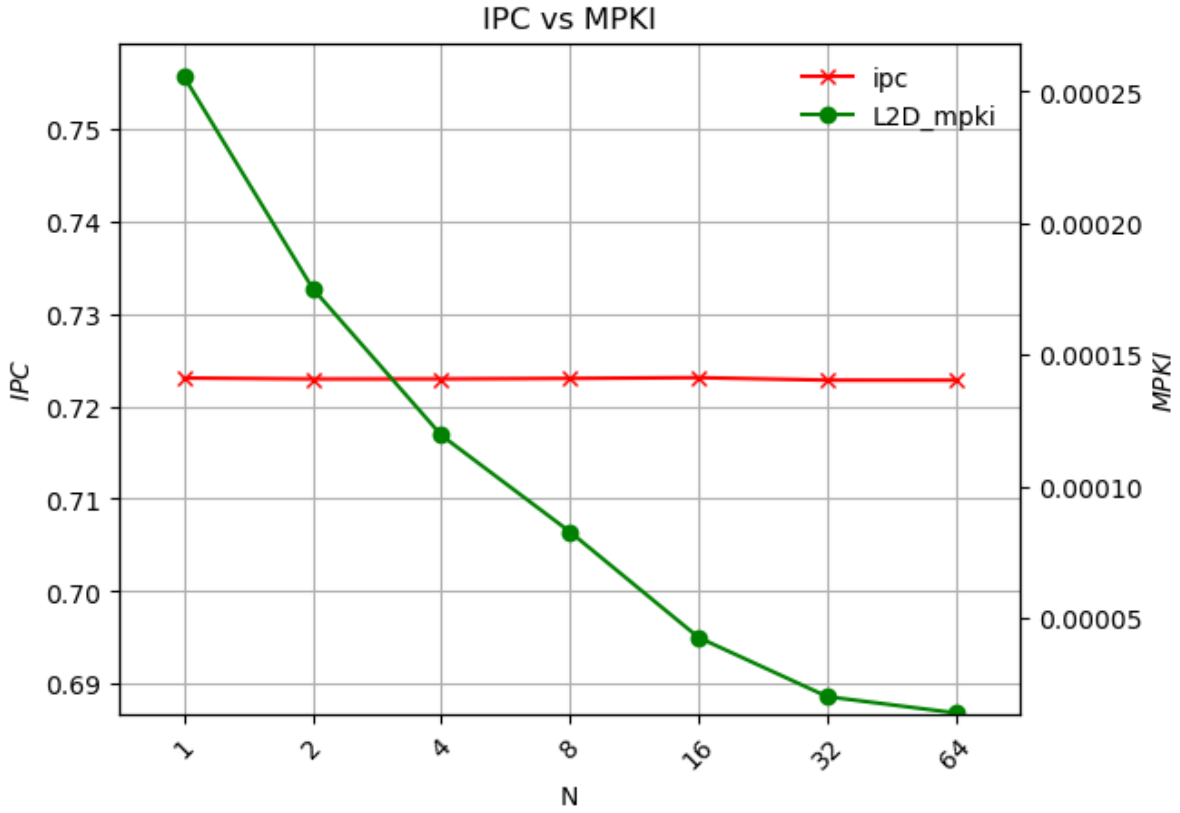
1.4.6 rtview


IPC vs MPKI
ipc
L2D_mpki
IPC
MPKI
N
0.77
0.76
0.75
0.74
0.73
0.72
0.71
0.70
0.09
0.08
0.07
0.06
0.05
1
2
4
8
16
32
64

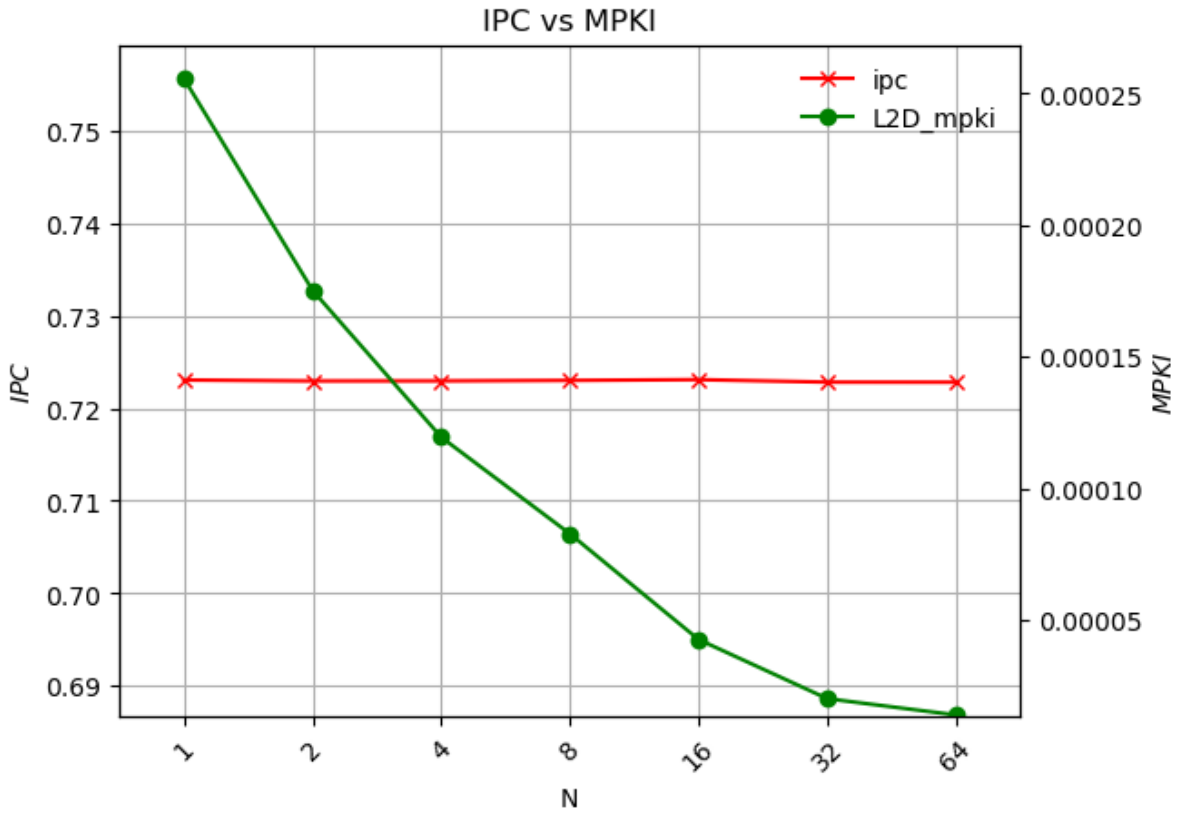
1.4.7 streamcluster



1.4.8 swaptions



1.4.9 Γεωμετρικός μέσος των benchmarks



1.4.10 Γενικές παρατηρήσεις για το prefetching στην L2

- Δεν φαίνεται να υπάρχει καθαρά καλύτερη επιλογή σε αυτή την περίπτωση (τουλάχιστον χρησιμοποιώντας σαν μετρική το IPC). Αν πραγματικά θέλουμε να διαλέξουμε την καλύτερη δυνατή επιλογή, θα χρησιμοποιήσουμε σαν μετρική το MPKI και θα επιλέξουμε N = 64, όπως είναι λογικό.
- Υπάρχουν αρκετά benchmarks που δεν επηρεάζονται σχεδόν καθόλου από το prefetching (bodytrack, canneal, rtview, swaptions), οπότε σε αυτά τα benchmarks μπορούμε να πούμε με σχετική σιγουριά ότι δεν γίνονται προσβάσεις σε διαχοδικά blocks.
- Στα υπόλοιπα benchmarks υπάρχει σχετική αύξηση (fluidanimate), ενώ στο streamcluster υπάρχει ραγδαία αύξηση όσο αυξάνεται το prefetching.
- Τέλος, το freqmine εμφανίζει μείωση για πιο επιθετικό prefetching, κάτι που μπορεί να οφείλεται στο ότι διώχνονται blocks που μπορεί να χρησιμοποιεί το πρόγραμμα για χάρη του prefetching.
- Συμπερασματικά, το prefetching είναι γενικά ωφέλιμο, μέχρι ενός ορίου.

2 Ζητούμενο 2ο

Επειδή στην πράξη οι τροποποιήσεις στην μικροαρχιτεκτονική του επεξεργαστή προκαλούν αλλαγές στην διάρκεια του κύκλου ρολογιού, κάνουμε την υπόθεση ότι ο διπλασιασμός του as-

sociativity προκαλεί αύξηση του κύκλου κατά 10%, ενώ ο διπλασιασμός του μεγέθους προκαλεί αύξηση 15%. Έτσι χρησιμοποιούμε ως αναφορά την πρώτη μέτρηση που εκτελέσαμε σε κάθε benchmark και αναλόγως μεταβάλλουμε τον κύκλο ρολογιού στις επόμενες μετρήσεις. Έτσι, αν ορίσουμε:

$$M = \frac{\text{Νέα τιμή associativity}}{\text{Παλιά τιμή associativity}}$$

και:

$$N = \frac{\text{Νέα τιμή cache/TLB size}}{\text{Παλιά τιμή cache/TLB size}}$$

τότε έχουμε:

$$\text{Νέα διάρκεια κύκλου ρολογιού} = 1.10^{\log_2 M} \times 1.15^{\log_2 N} \times \text{Παλιά διάρκεια κύκλου ρολογιού}$$

και άρα:

$$\text{Νέο IPC} = \frac{\text{Παλιό IPC}}{1.10^{\log_2 M} \times 1.15^{\log_2 N}}$$

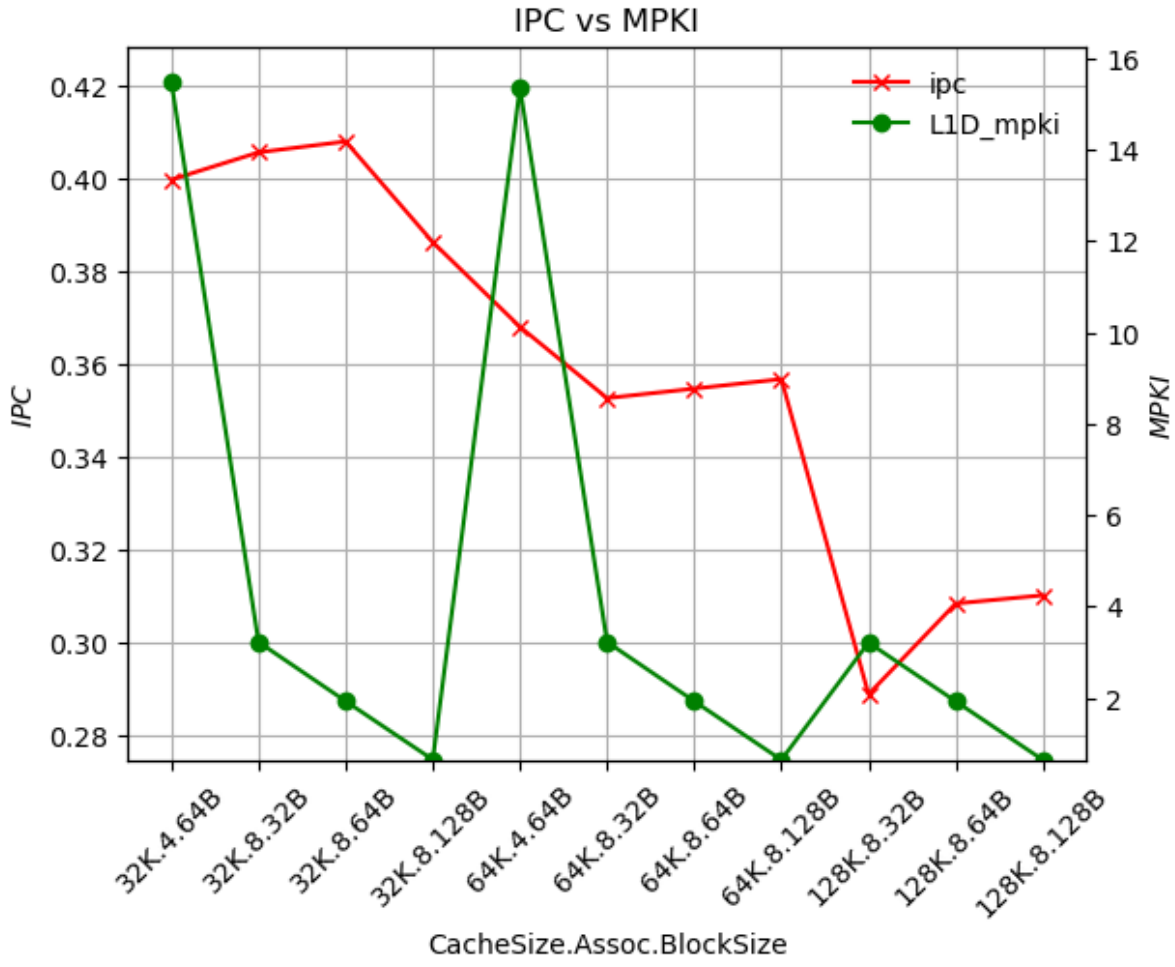
Τα παραπάνω αποτελέσματα ισχύουν για όλες τις τιμές των $M, N \in \{..., 2^{-3}, 2^{-2}, 2^{-1}, 1, 2, 2^2, 2^3, ...\}$, αφού:

- αν $M > 1$ ή $N > 1$, τότε το IPC μικραίνει, όπως είναι λογικό (μεγαλώνουν τα associativity και cache size),
- αν $\mathrm{M} < 1$ ή $\mathrm{N} < 1$, τότε οι λογάριθμοι παίρνουν αρνητική τιμή και το IPC μεγαλώνει (πάλι αναμενόμενο, αφού μικραίνουν τα associativity και cache size),
- αν $M = 1$ ή $N = 1$, τότε οι λογάριθμοι είναι μηδενικοί και δεν έχουμε καμία αλλαγή.

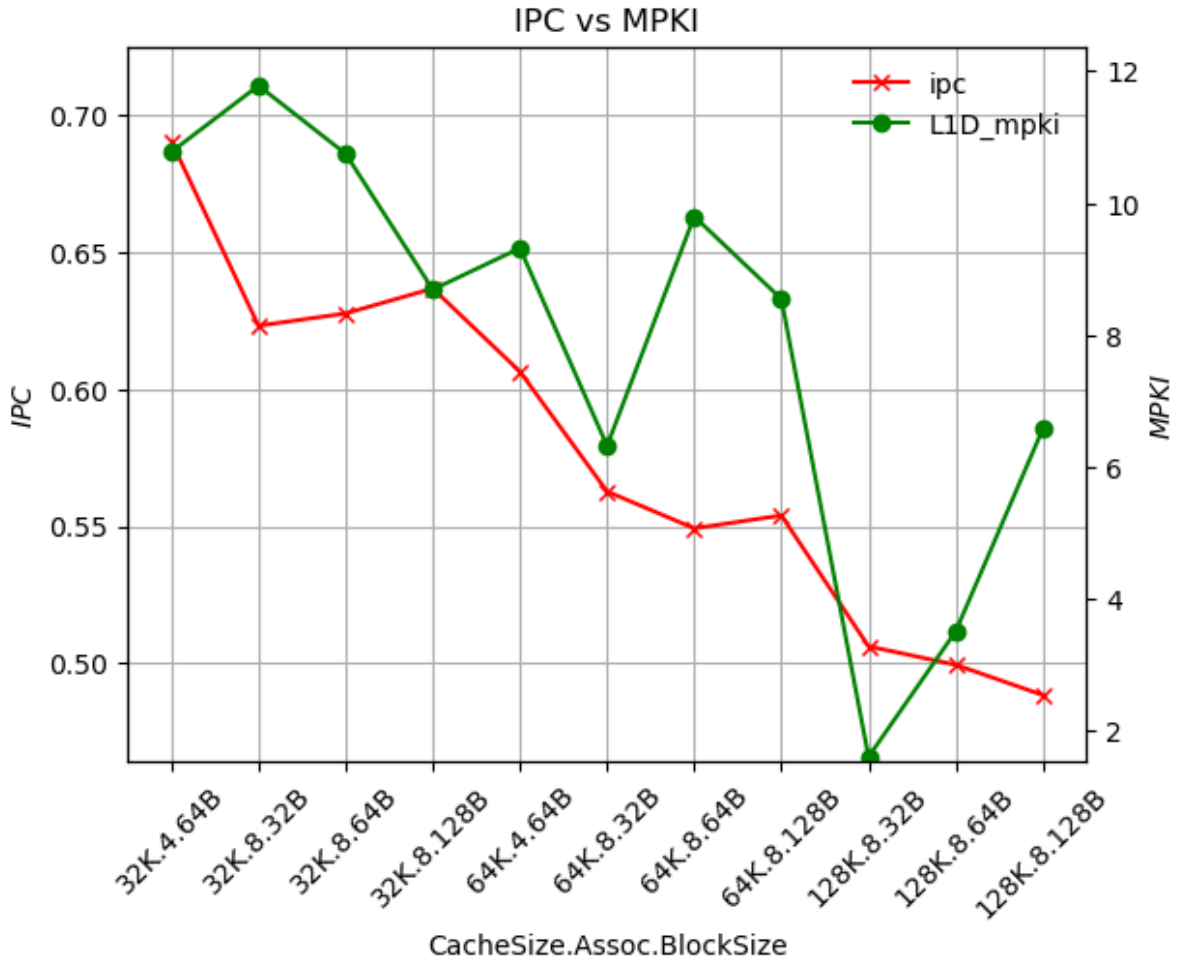
Με βάση την παραπάνω ανάλυση έχουμε:

2.1 L1 cache

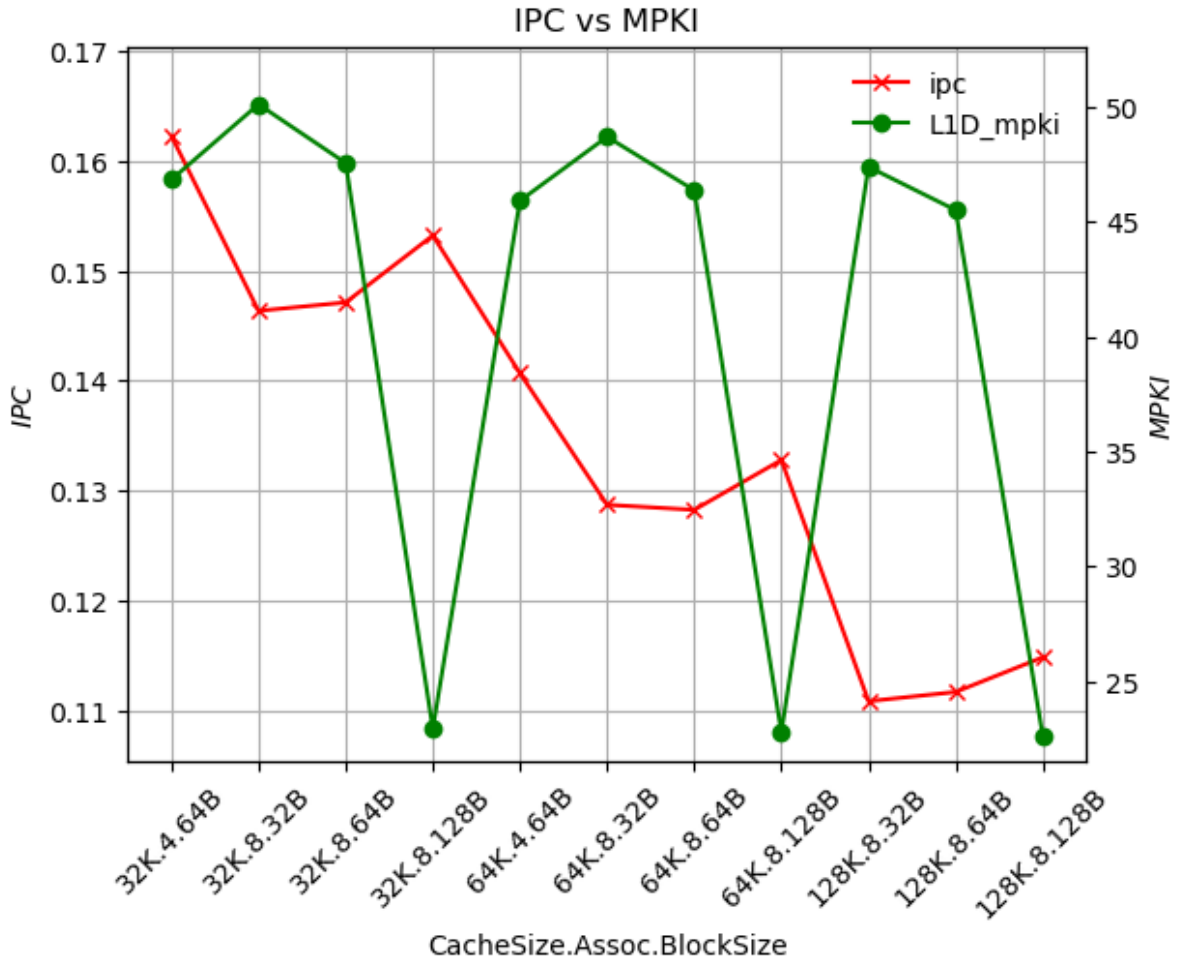
2.1.1 blackscholes (clock cycle architecture-dependent)



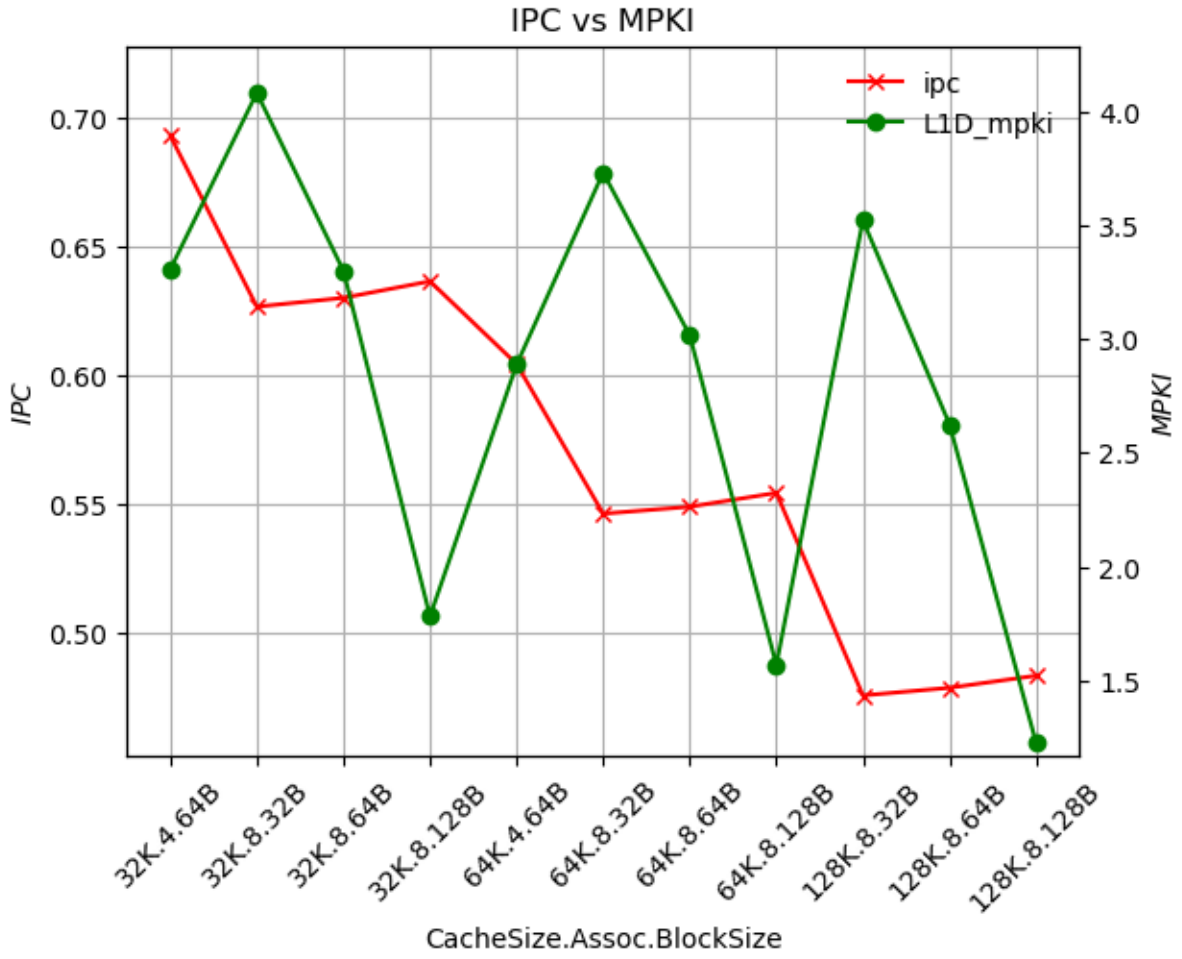
2.1.2 bodytrack (clock cycle architecture-dependent)



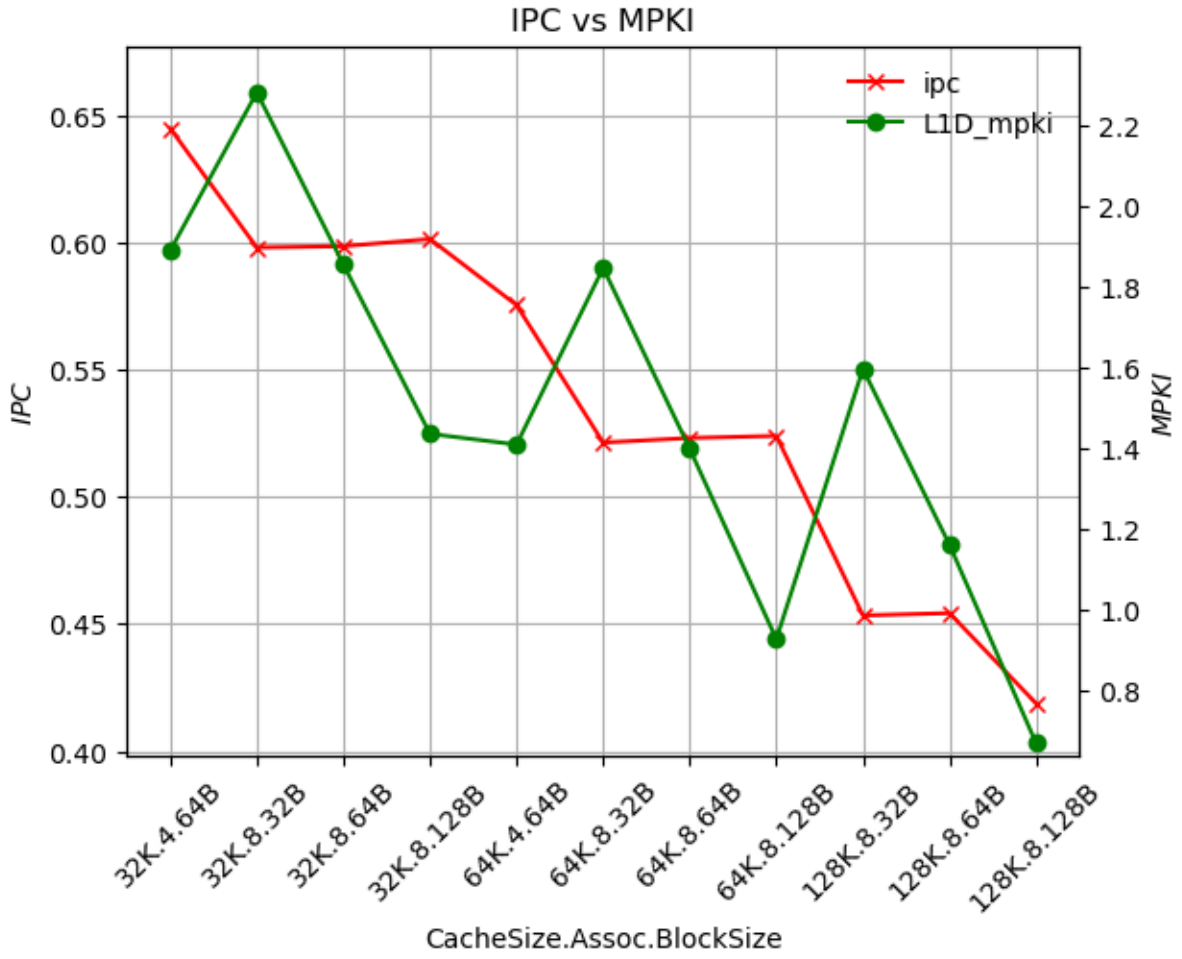
2.1.3 canneal (clock cycle architecture-dependent)



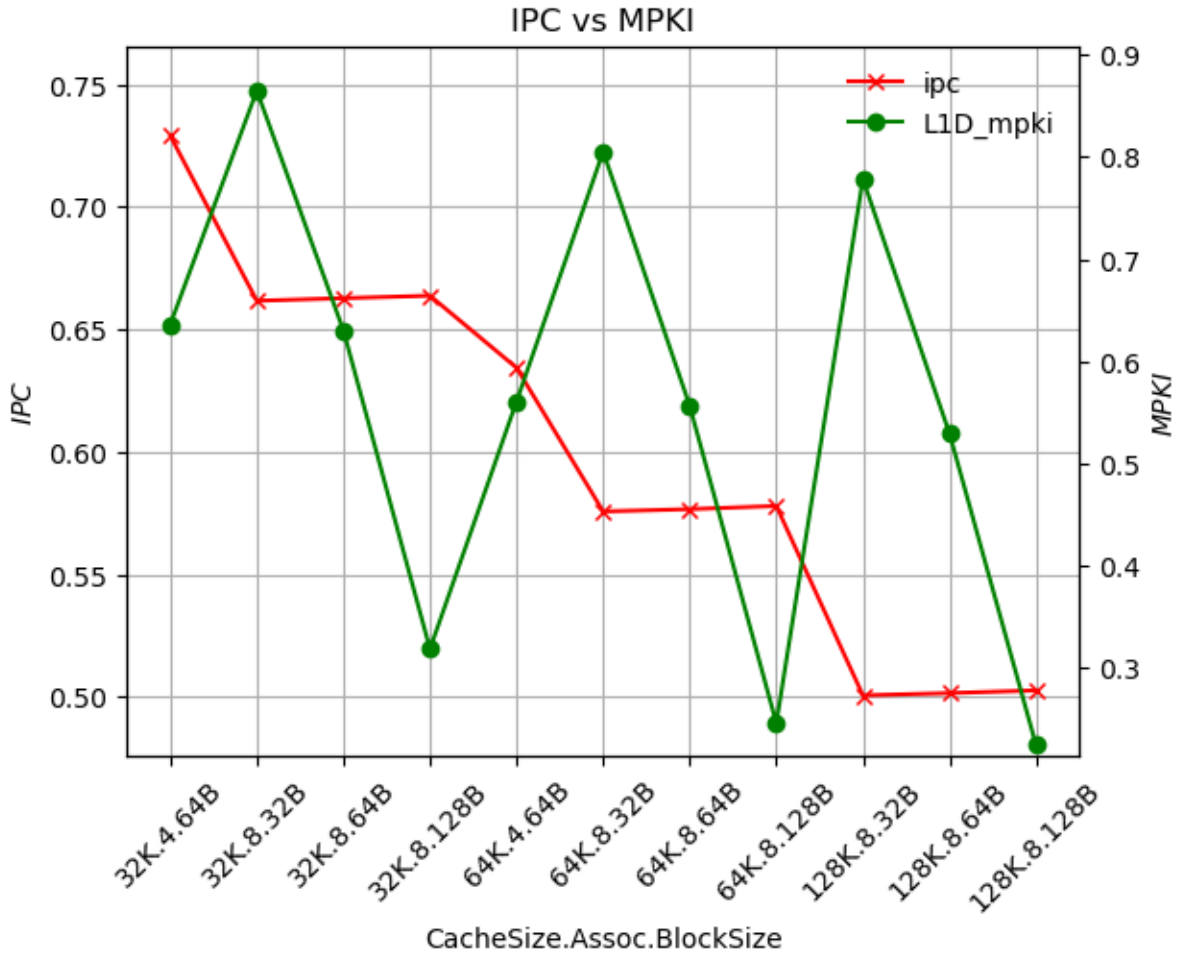
2.1.4 fluidanimate (clock cycle architecture-dependent)



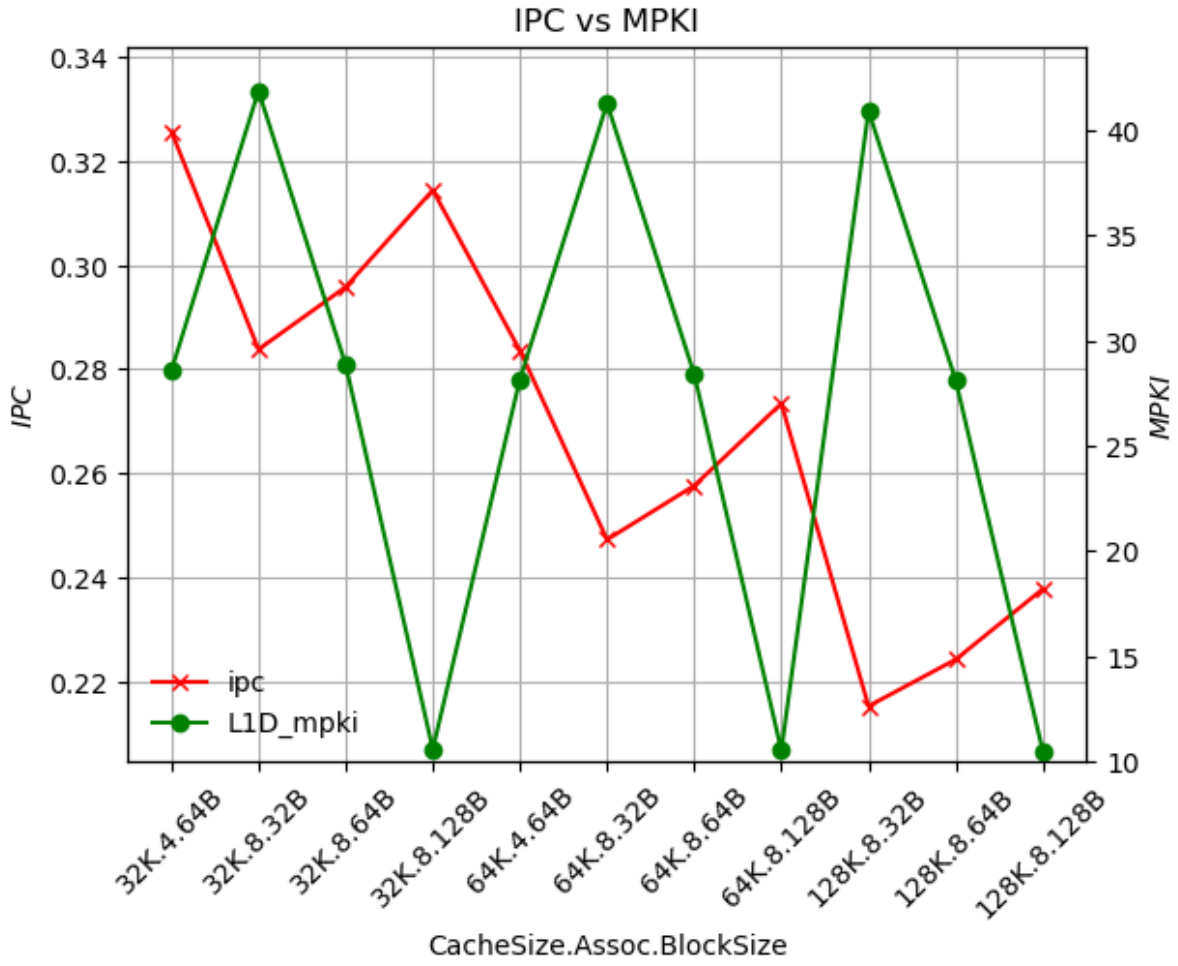
2.1.5 freqmine (clock cycle architecture-dependent)



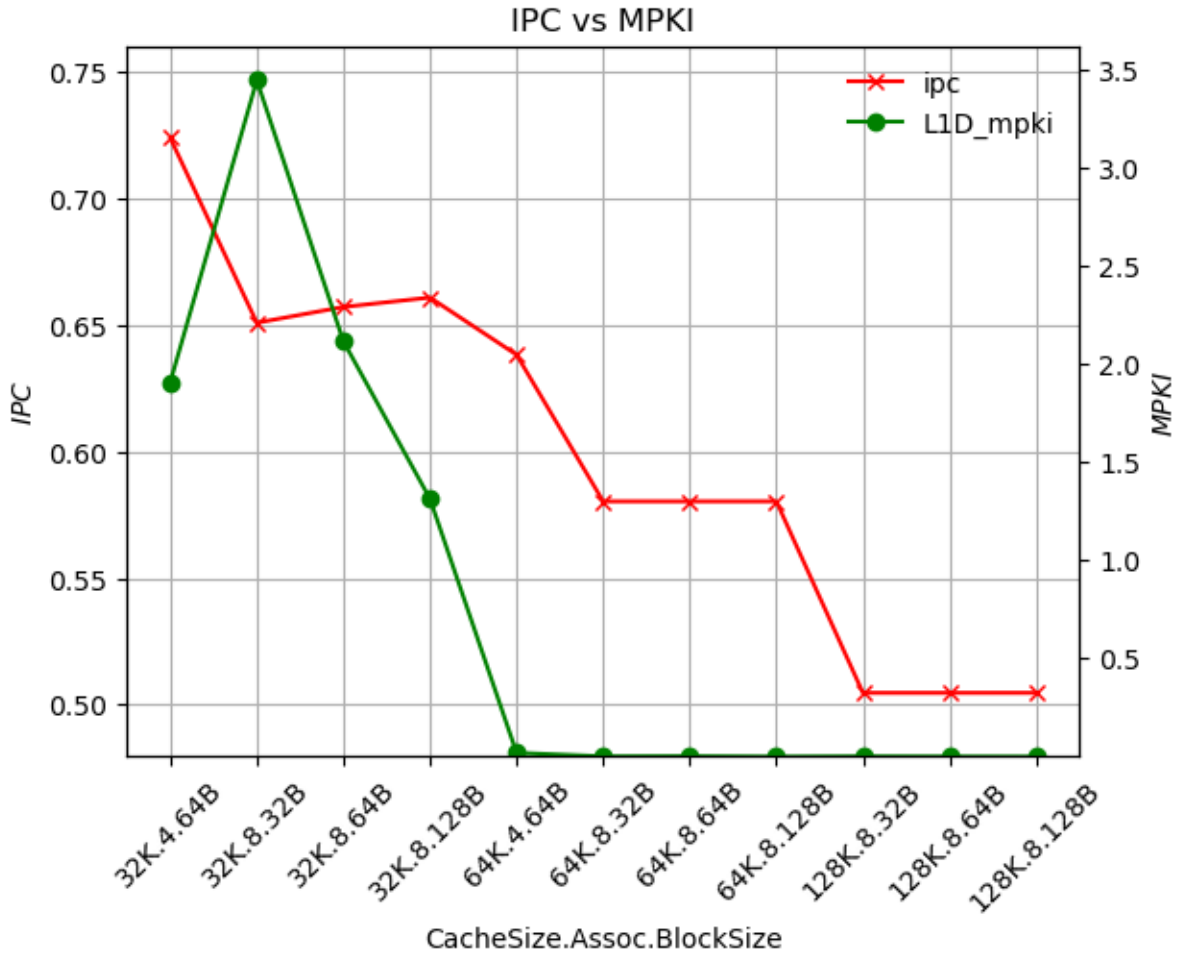
2.1.6 rtview (clock cycle architecture-dependent)



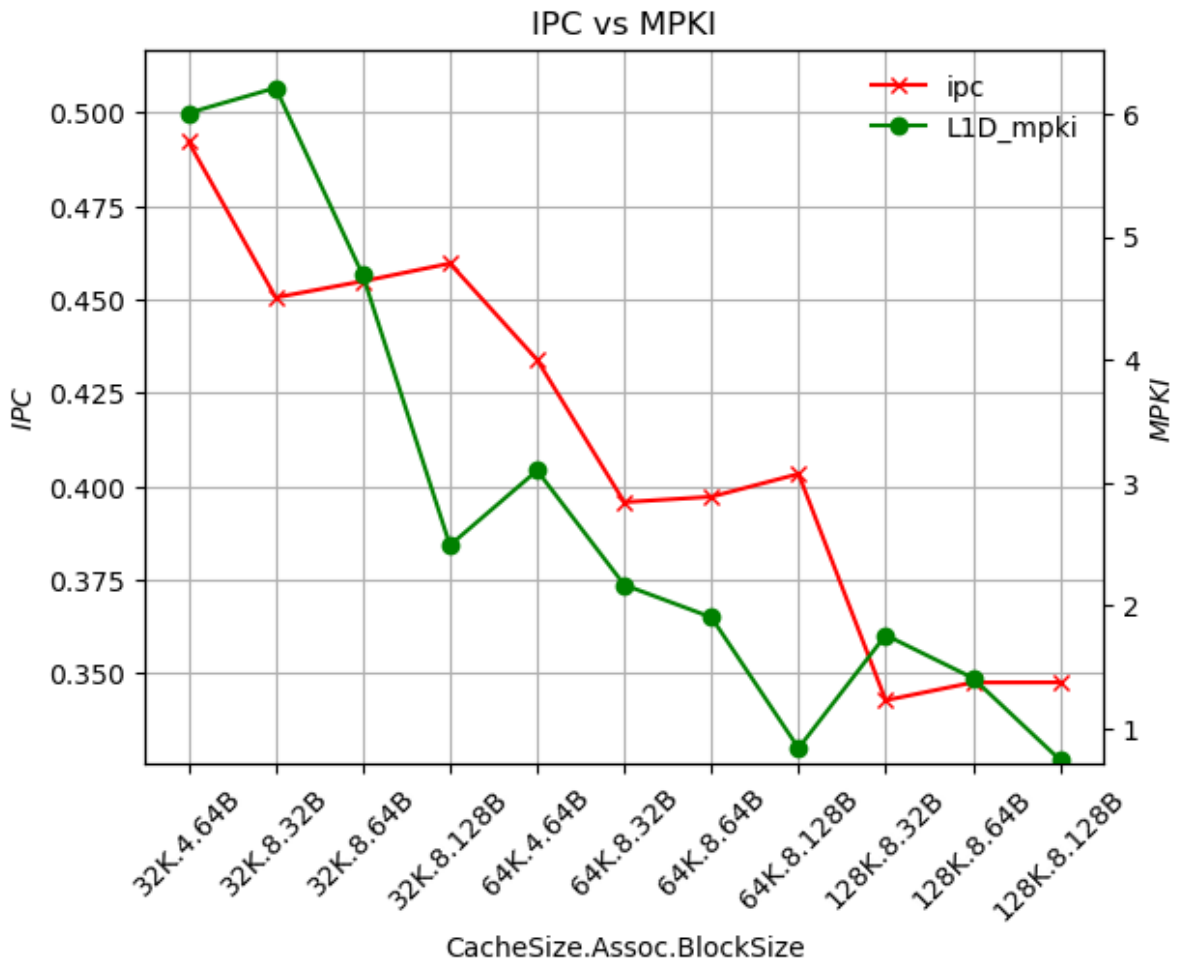
2.1.7 streamcluster (clock cycle architecture-dependent)



2.1.8 swaptions (clock cycle architecture-dependent)



2.1.9 Γεωμετρικός μέσος των benchmarks



2.1.10 Γενικές παρατηρήσεις για την L1 (κύκλος ρολογιού μεταβλητός)

- Καλύτερη επιλογή φαίνεται να είναι η τριπλέτα:

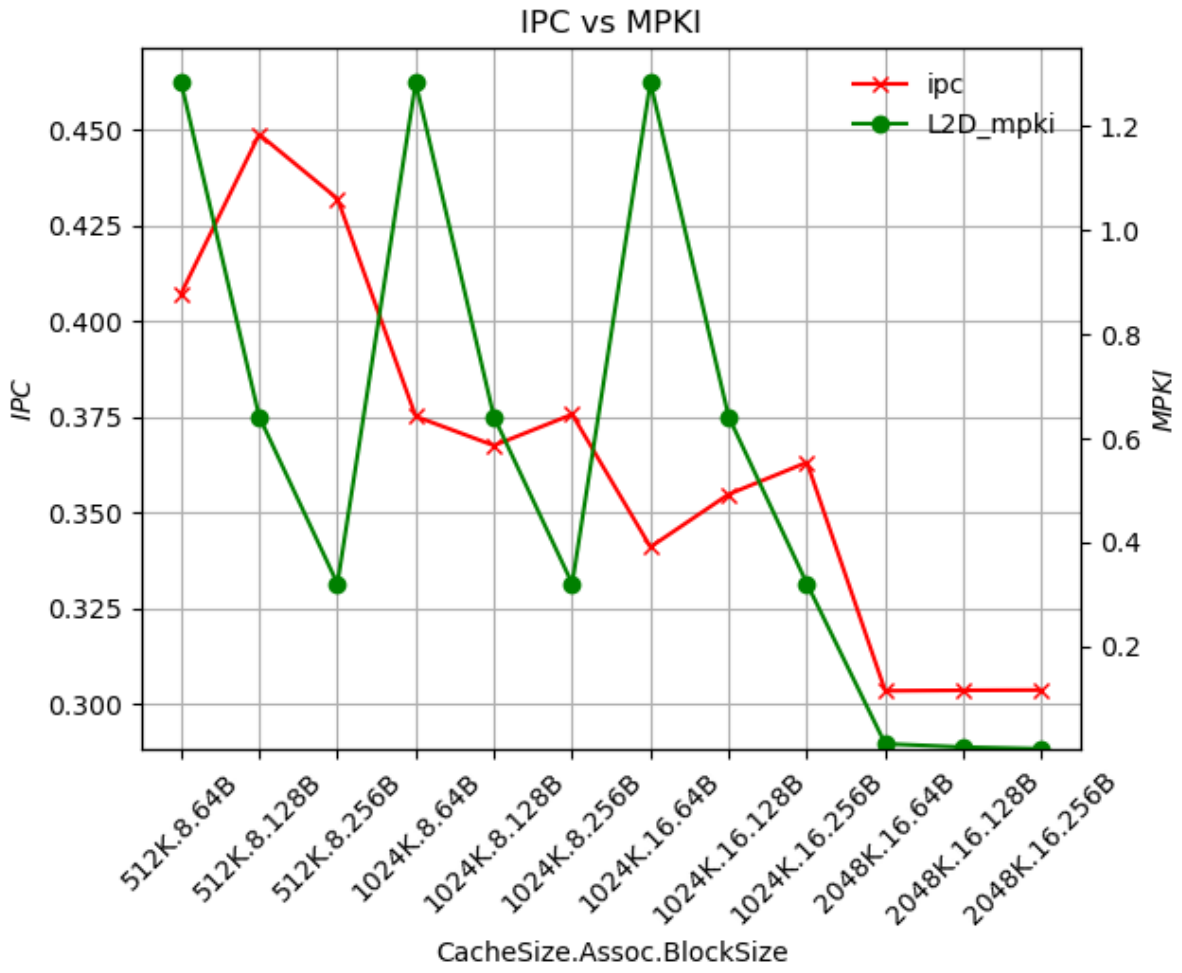
  `(cache size, associativity, block size) = (32K, 4, 64B)`

- Πλέον τα IPC και MPKI δεν είναι αντιστρόφως ανάλογα, αφού για να έχουμε λιγότερα misses πρέπει να συμβιβαστούμε με μεγαλύτερους κύκλους ρολογιού.
- Σε όλες τις περιπτώσεις η αύξηση των μεγεθών που δημιουργούν το επιπλέον overhead στον κύκλο ρολογιού, δηλαδή cache size και associativity, επιδρά πλέον αρνητικά στο IPC.
- Το μόνο μέγεθος που είτε είναι ωφέλιμο είτε δεν επιδρά καθόλου είναι το μέγεθος του block size, αφού δεν έχει επίπτωση τον κύκλο ρολογιού.

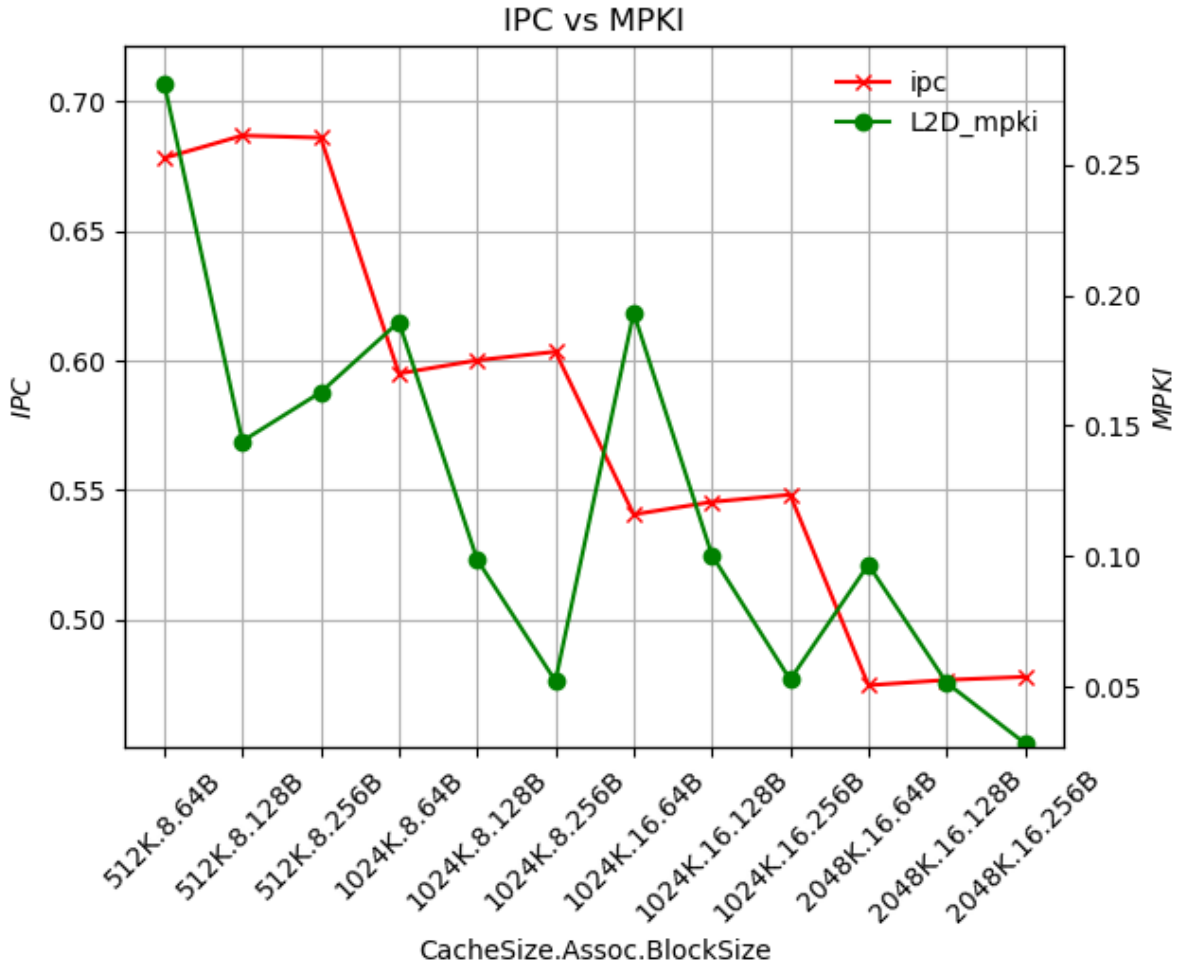
Φαίνεται ότι όταν οι συνθήκες γίνονται πιο ρεαλιστικές (ώστε ο κύκλος ρολογιού να εξαρτάται από τα μικροαρχιτεκτονικά χαρακτηριστικά του επεξεργαστή), τα οφέλη των αυξημένων παραμέτρων (cache size, associativity, block size) ”κοστίζουν” ακριβά στο IPC, και παρά την μείωση του MPKI η επίδοση χειροτερεύει.

2.2 L2 cache

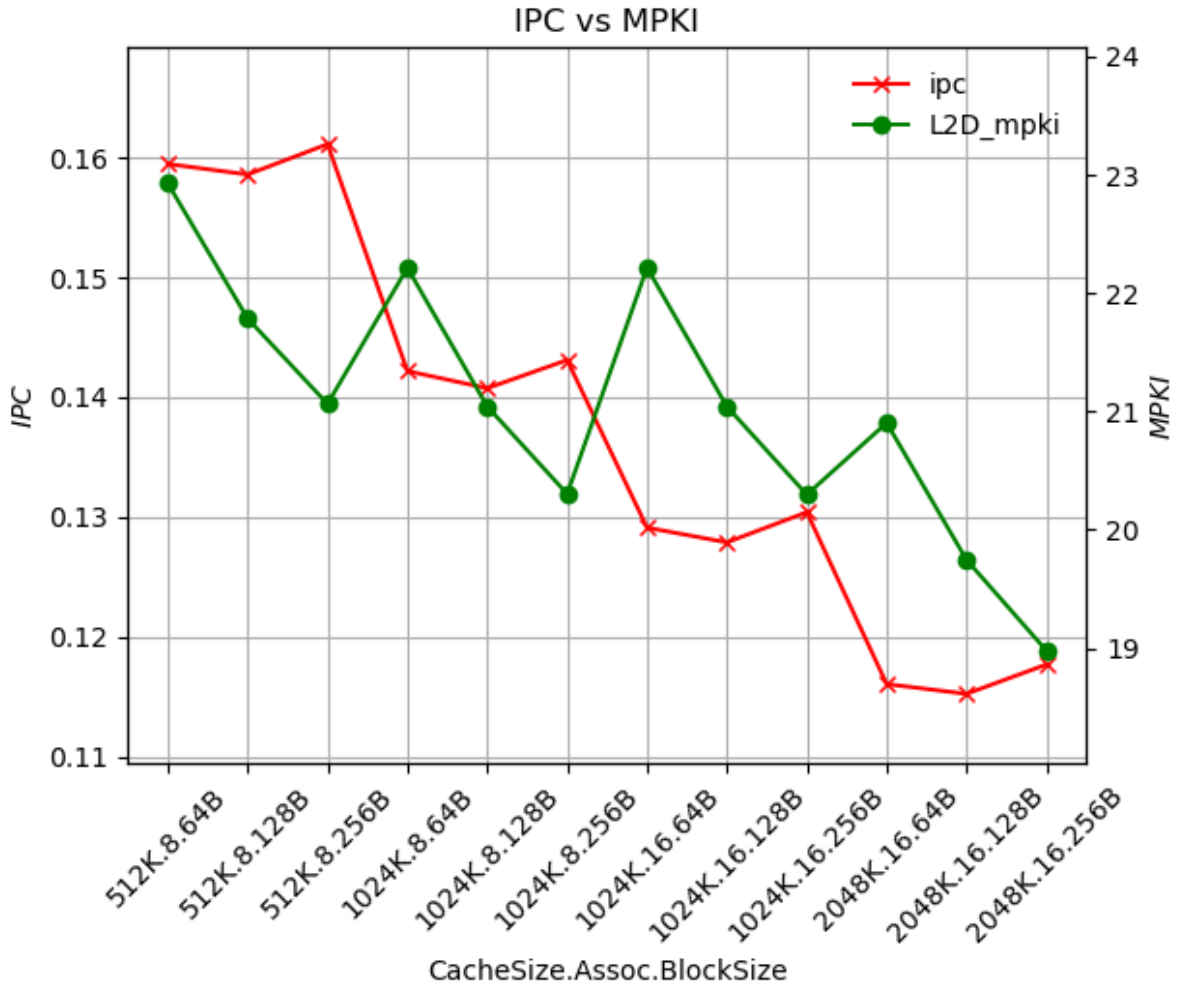
2.2.1 blackscholes (clock cycle architecture-dependent)



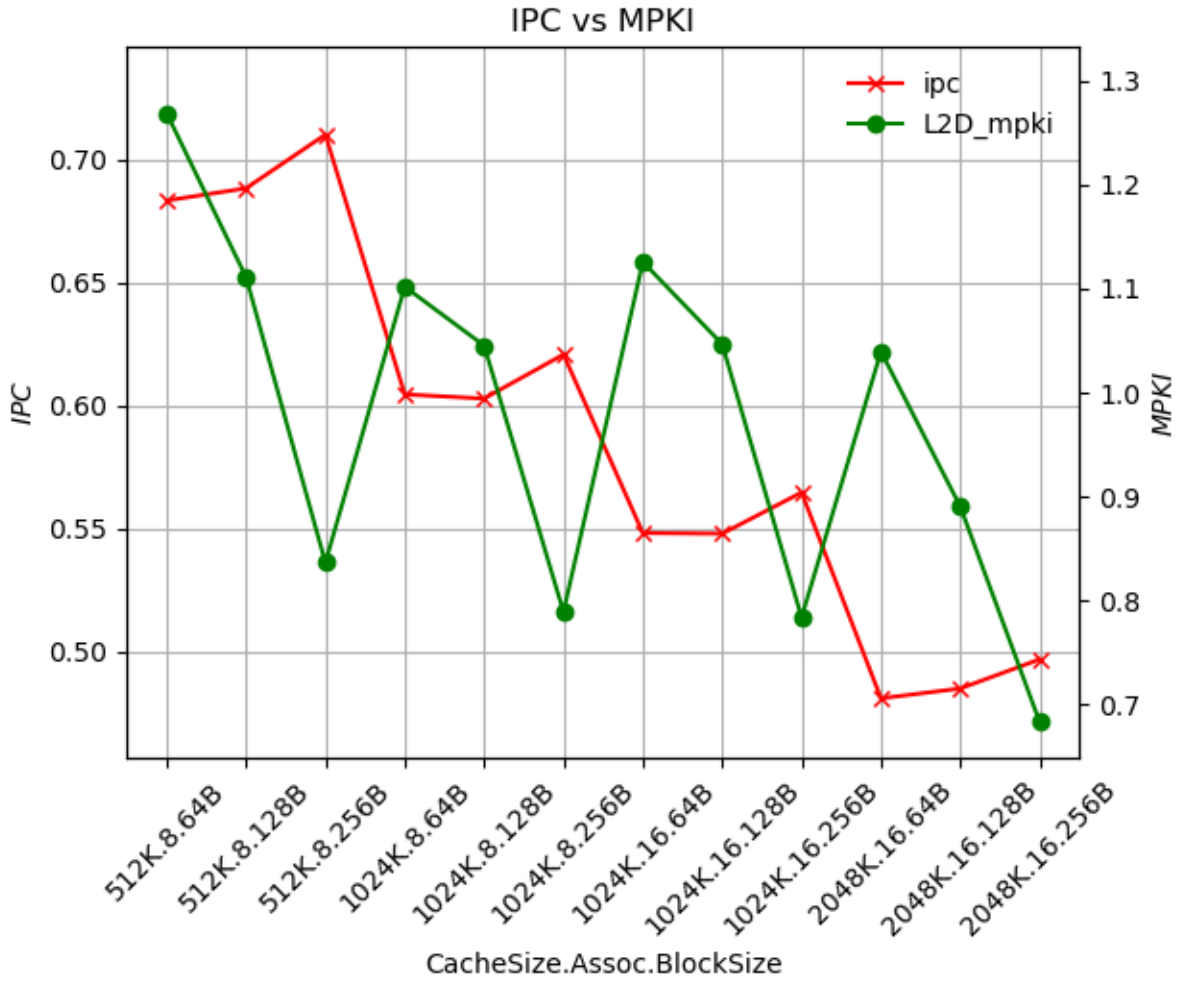
2.2.2 bodytrack (clock cycle architecture-dependent)



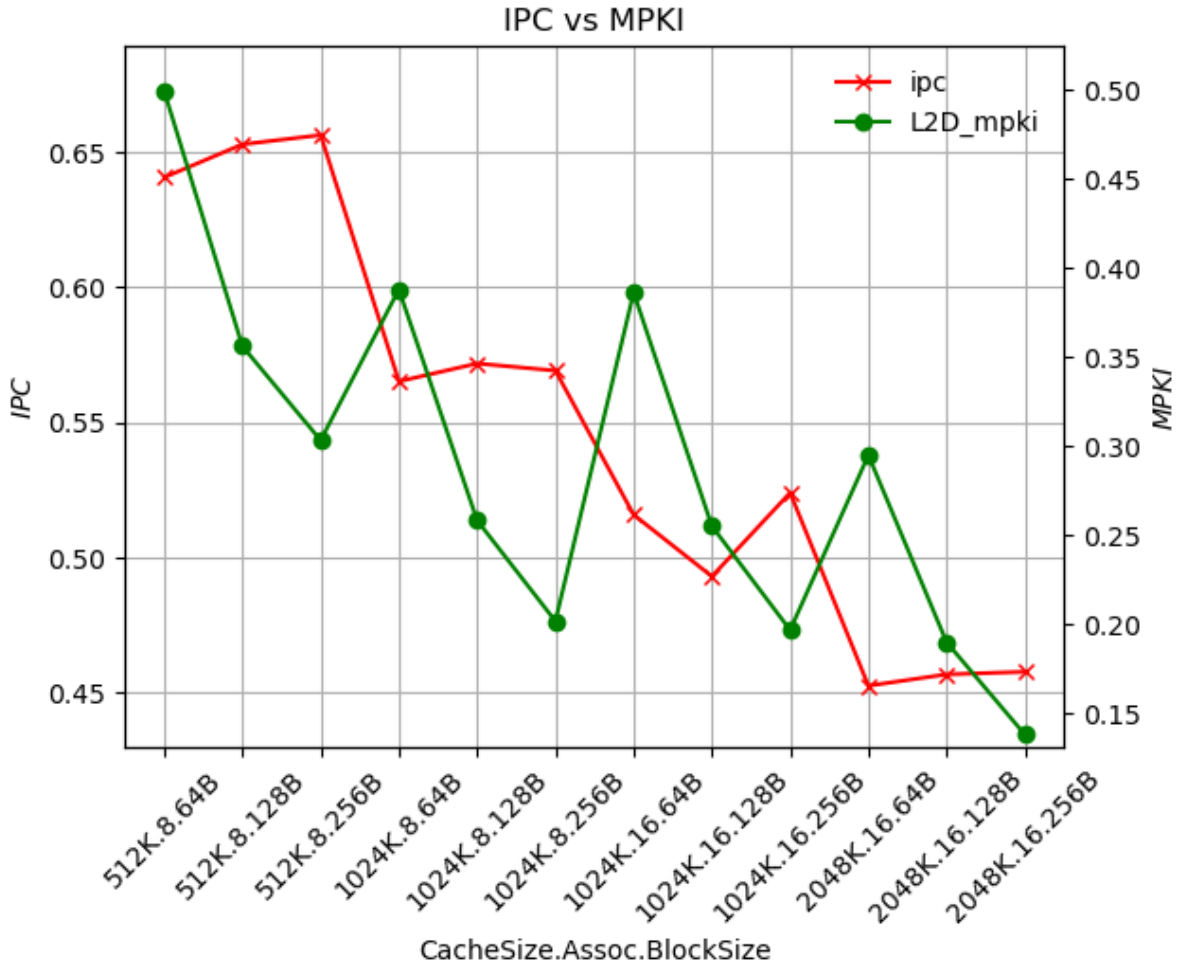
2.2.3 canneal (clock cycle architecture-dependent)



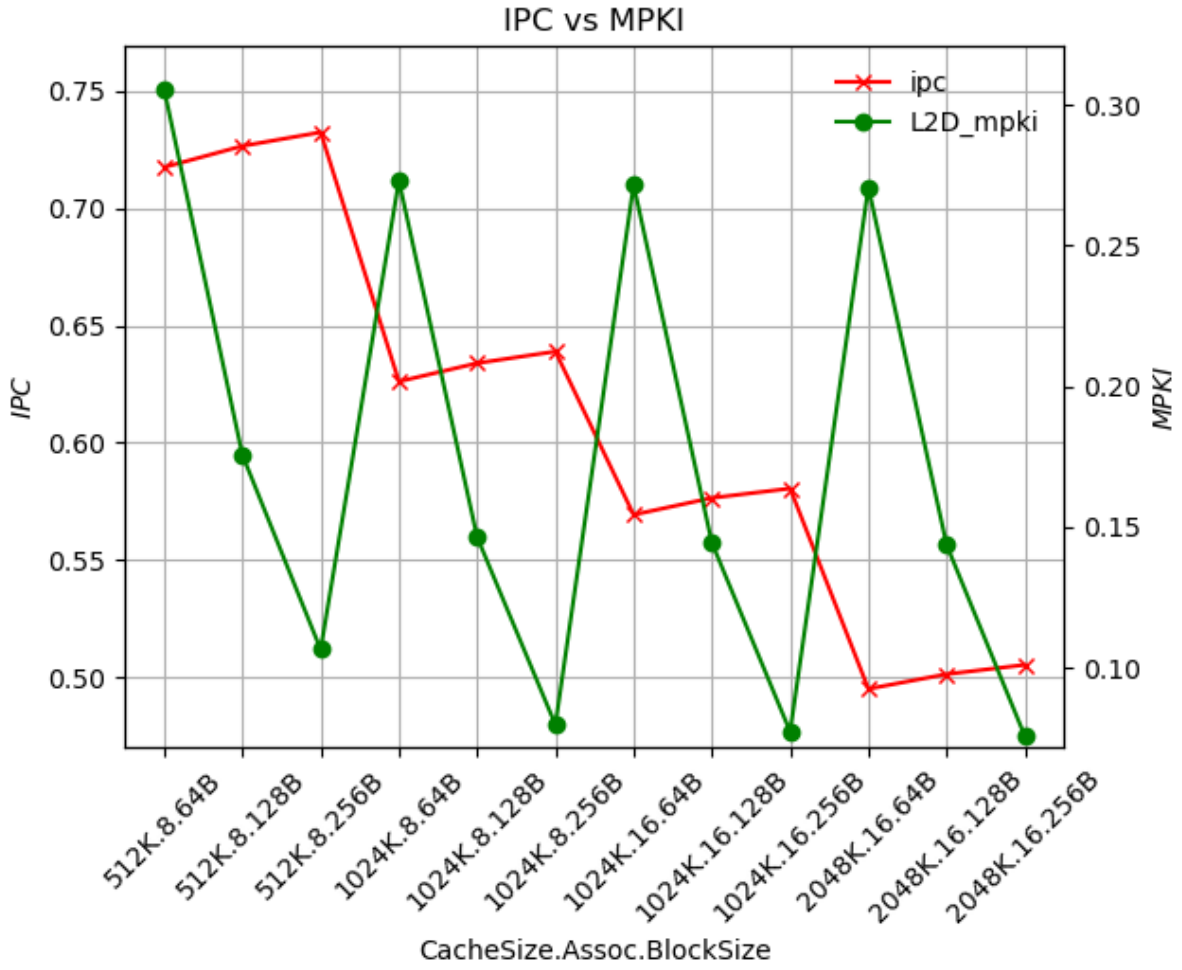
2.2.4 fluidanimate (clock cycle architecture-dependent)



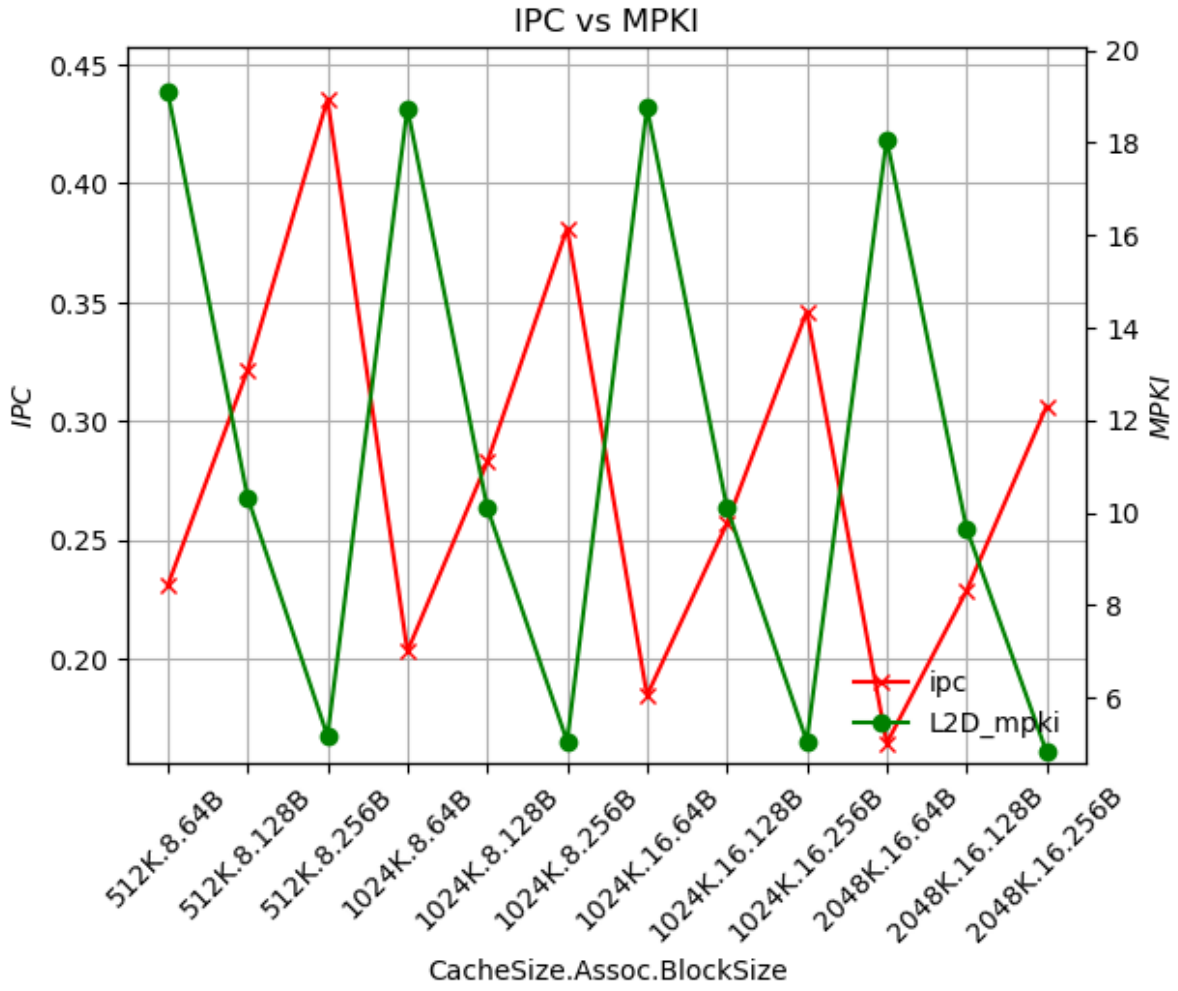
2.2.5 freqmine (clock cycle architecture-dependent)



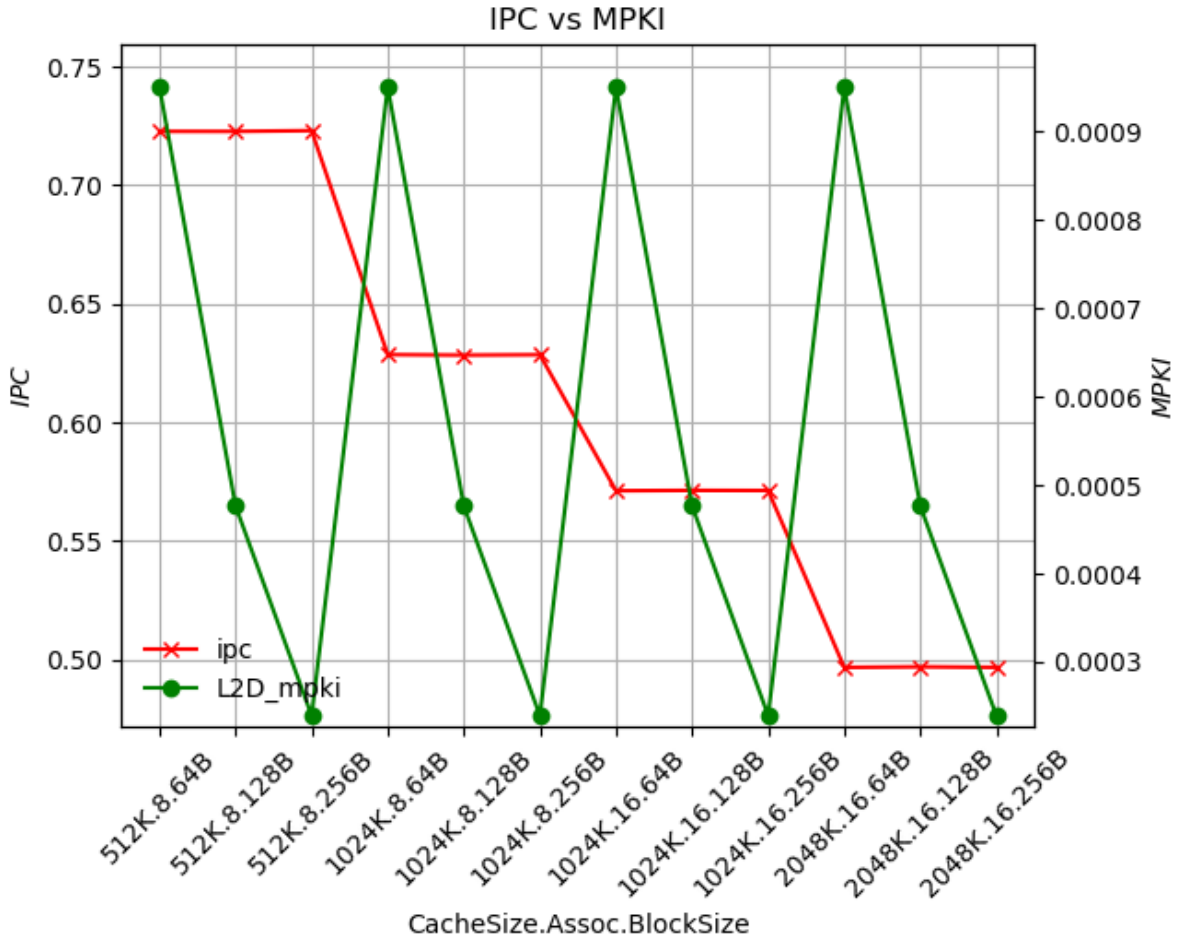
2.2.6 rtview (clock cycle architecture-dependent)



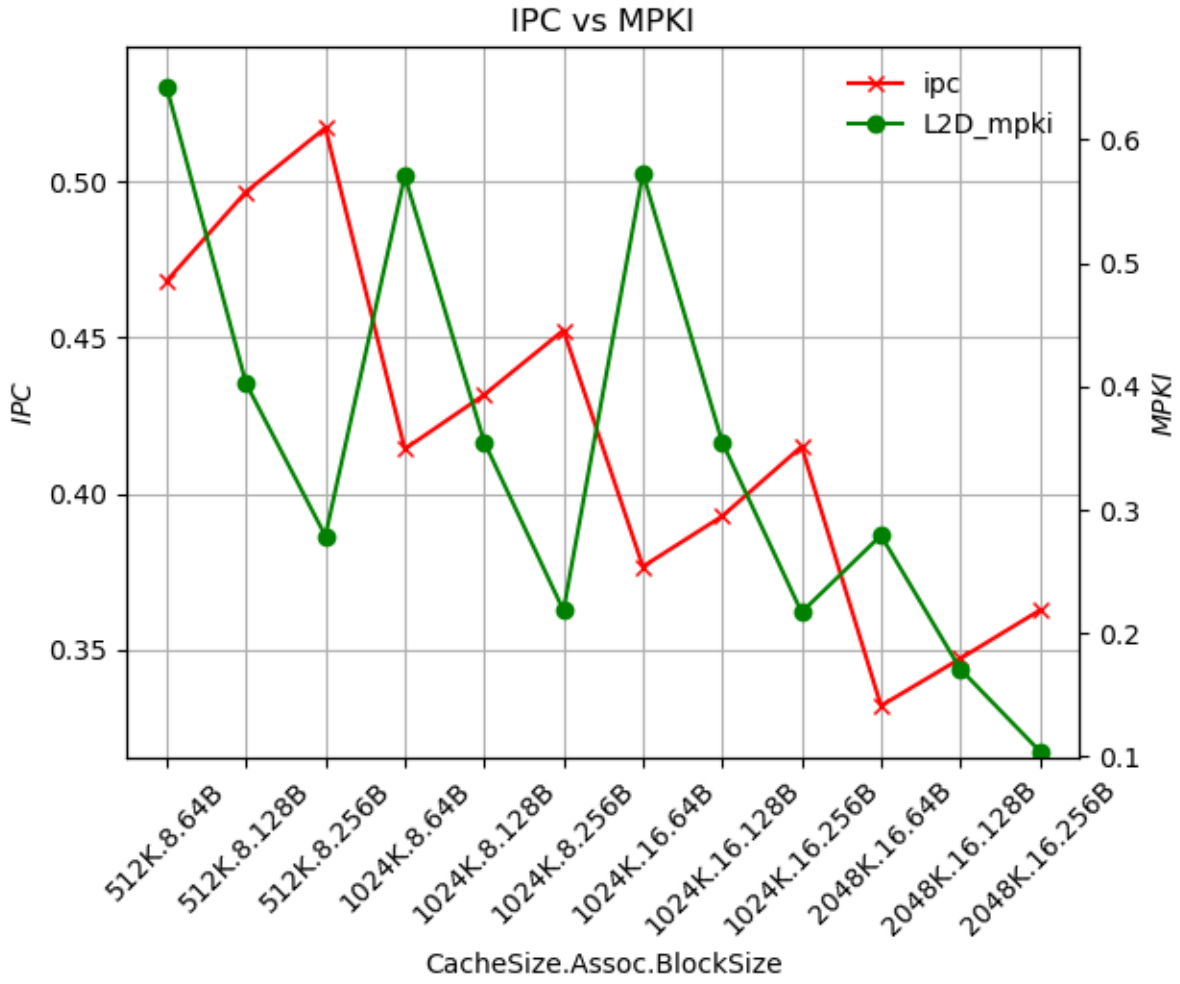
2.2.7 streamcluster (clock cycle architecture-dependent)



2.2.8 swaptions (clock cycle architecture-dependent)



2.2.9 Γεωμετρικός μέσος των benchmarks



2.2.10 Γενικές παρατηρήσεις για την L2 (κύκλος ρολογιού μεταβλητός)

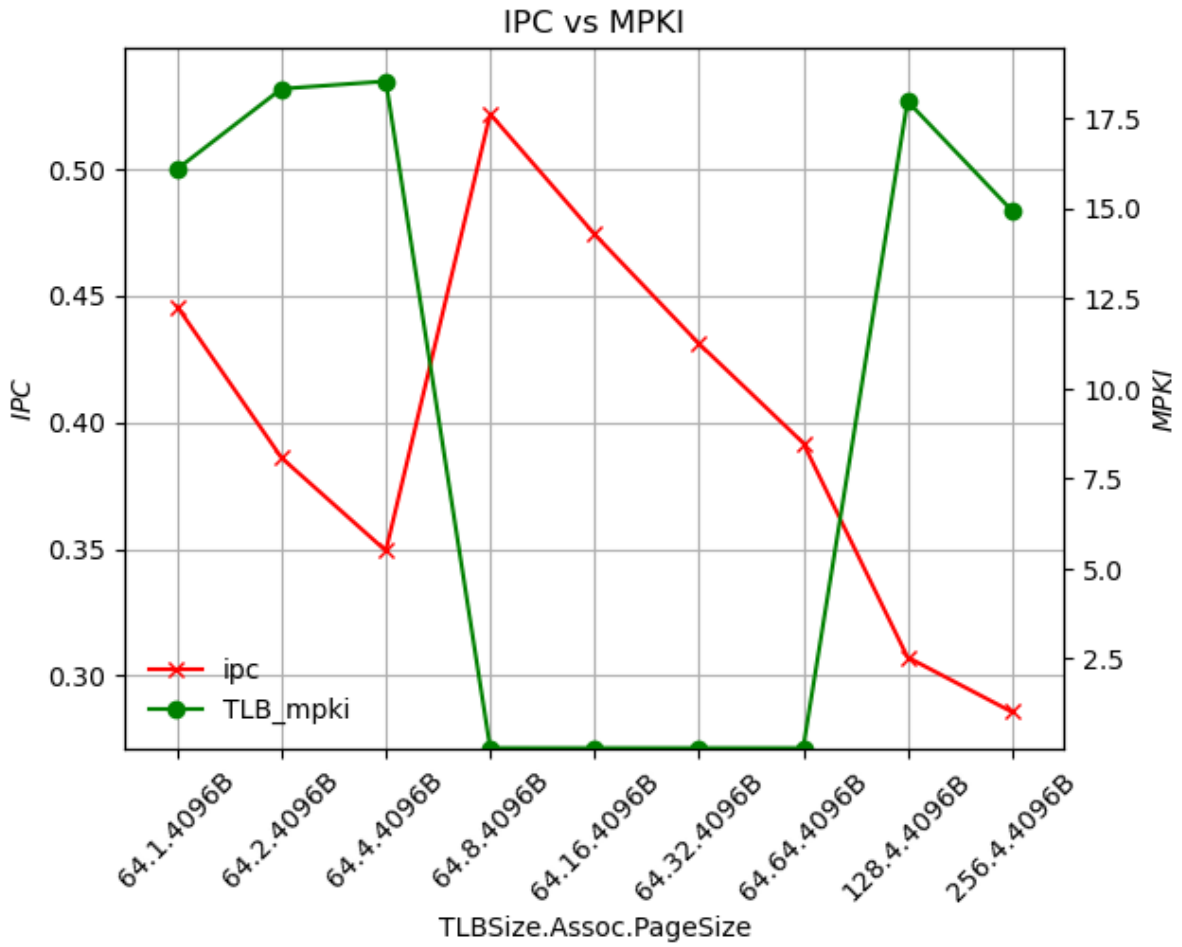
- Η καλύτερη επιλογή φαίνεται να είναι η τριπλέτα:

  `(cache size, associativity, block size) = (512K, 8, 256B)`

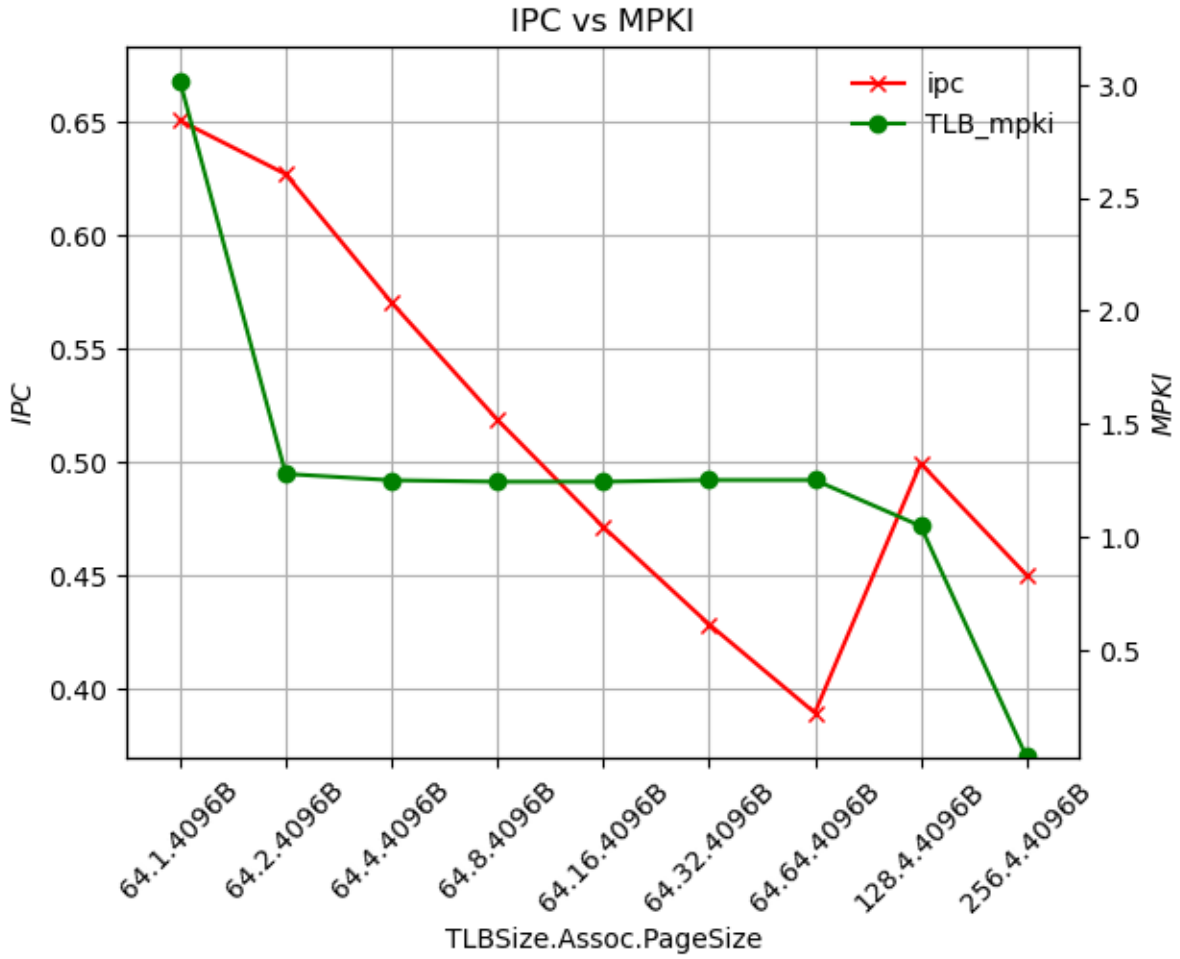
- Εδώ ισχύουν τα ίδια με την L1, απλώς εδώ η αύξηση του block size μας δίνει αρκετά μεγαλύτερο όφελος.

2.3 TLB

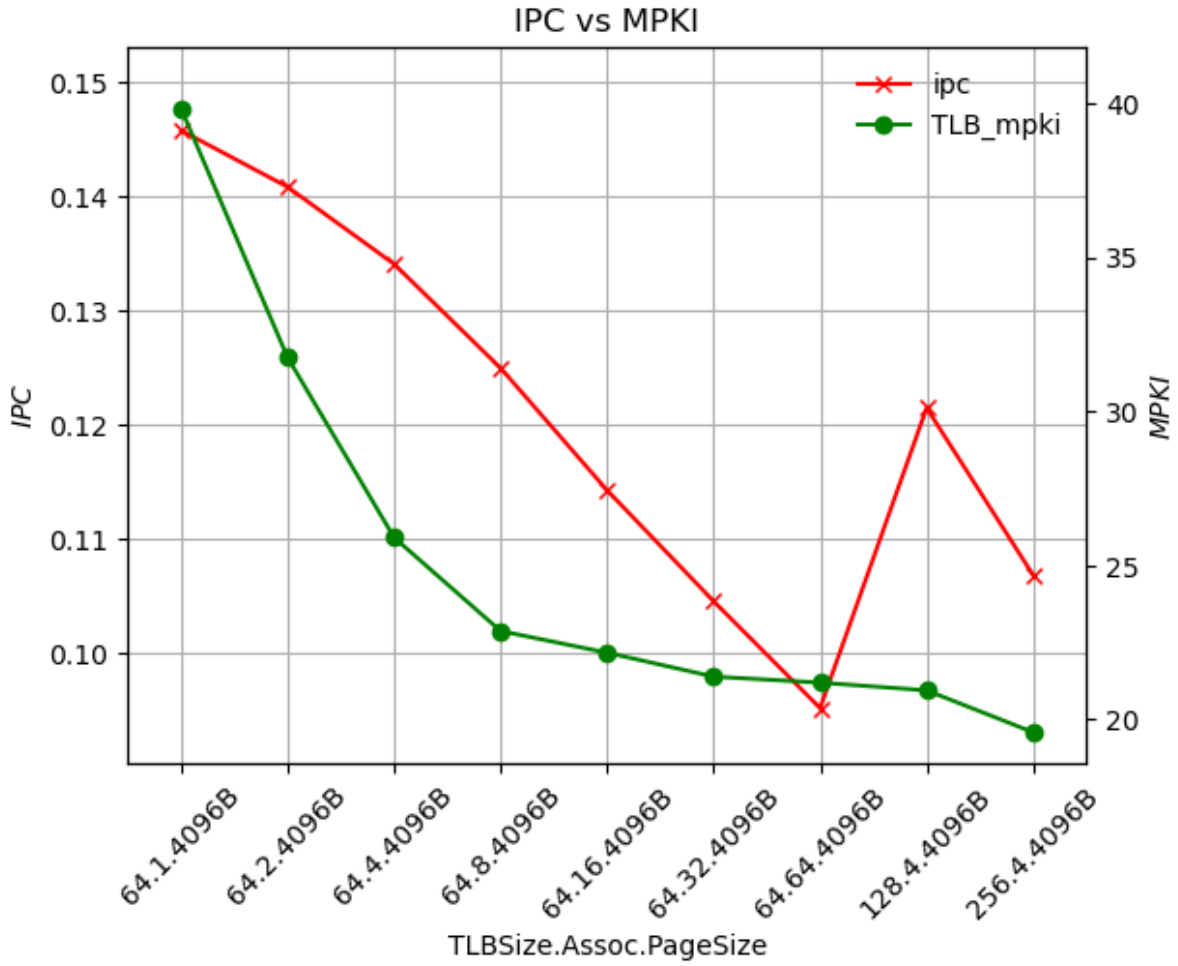
2.3.1 blackscholes (clock cycle architecture-dependent)



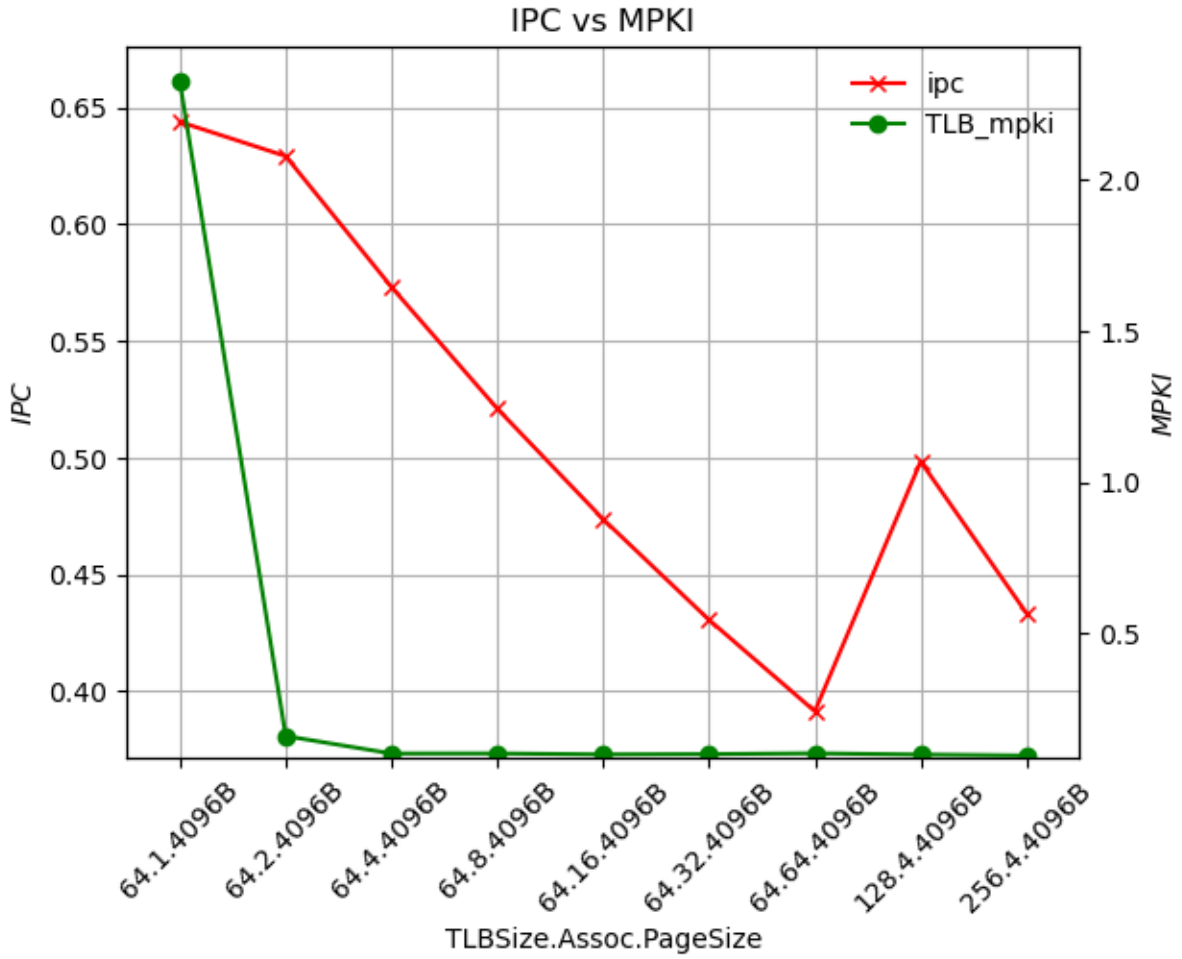
2.3.2 bodytrack (clock cycle architecture-dependent)



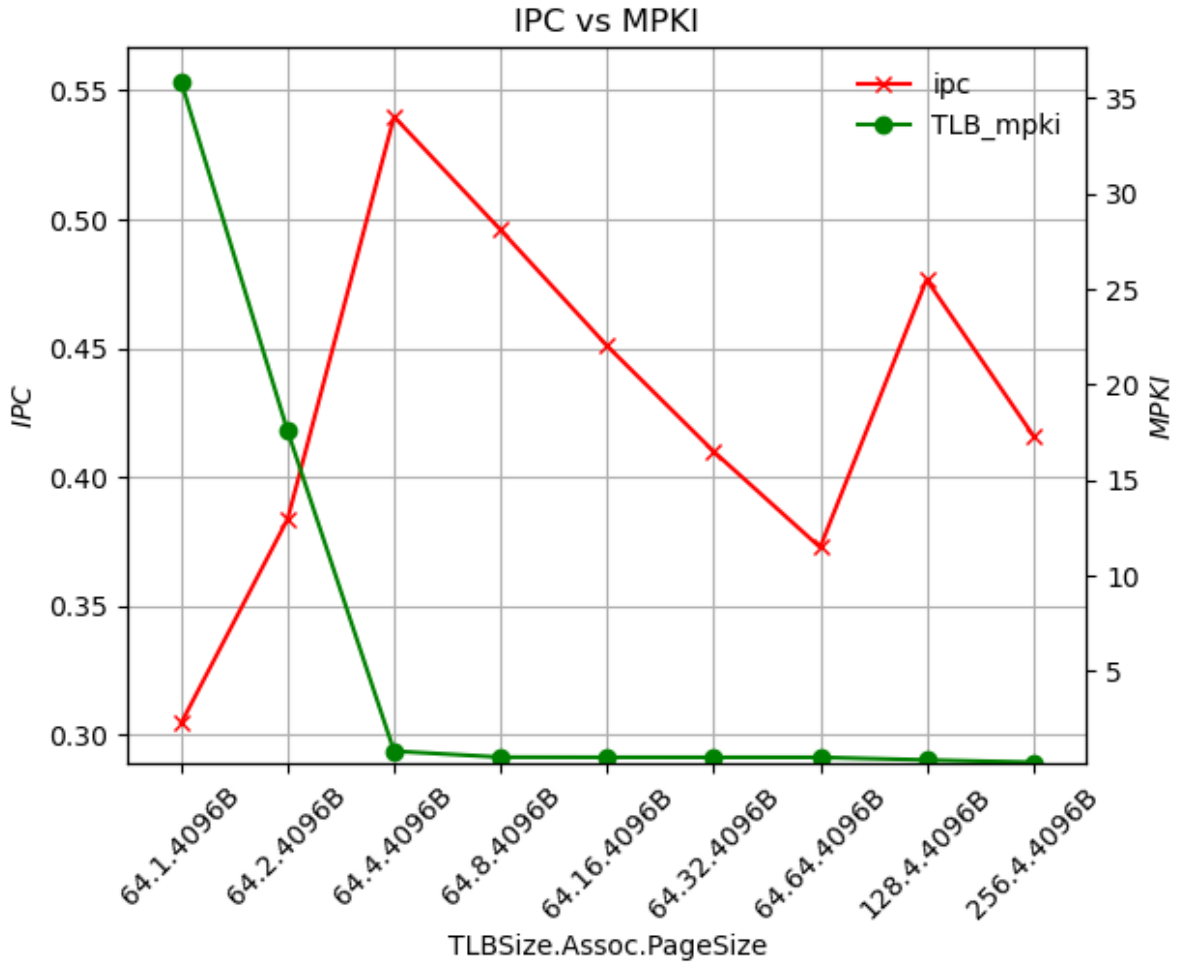
2.3.3 canneal (clock cycle architecture-dependent)



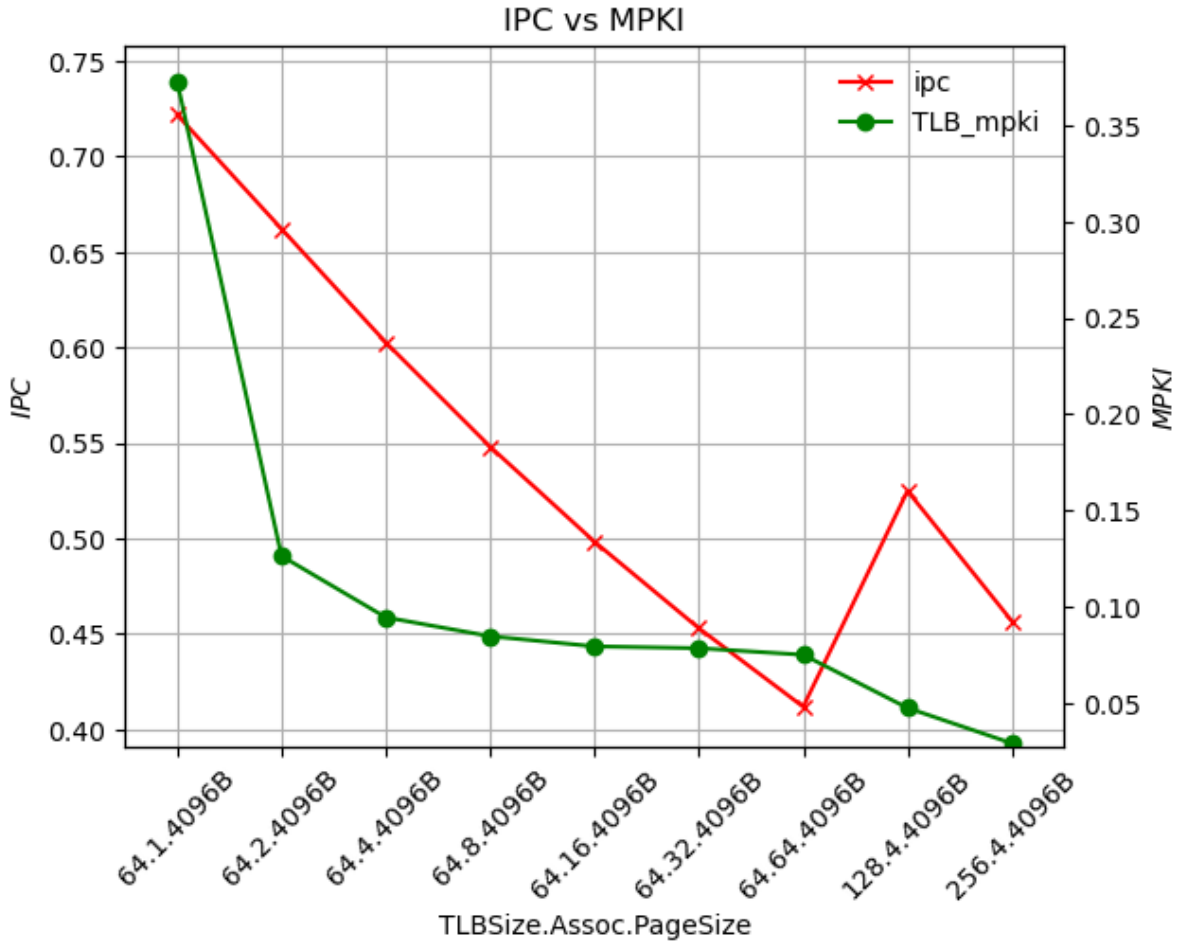
2.3.4 fluidanimate (clock cycle architecture-dependent)



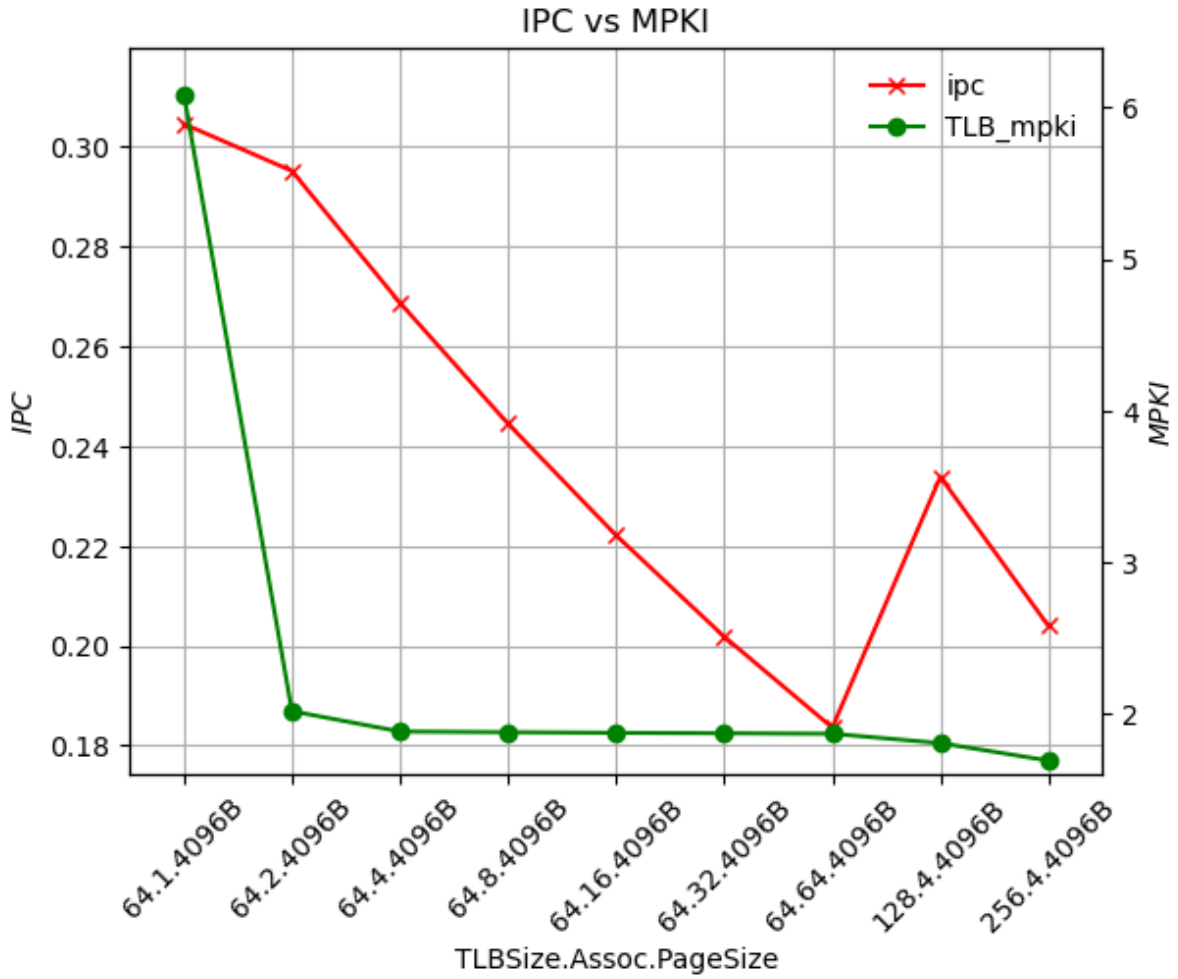
2.3.5 freqmine (clock cycle architecture-dependent)



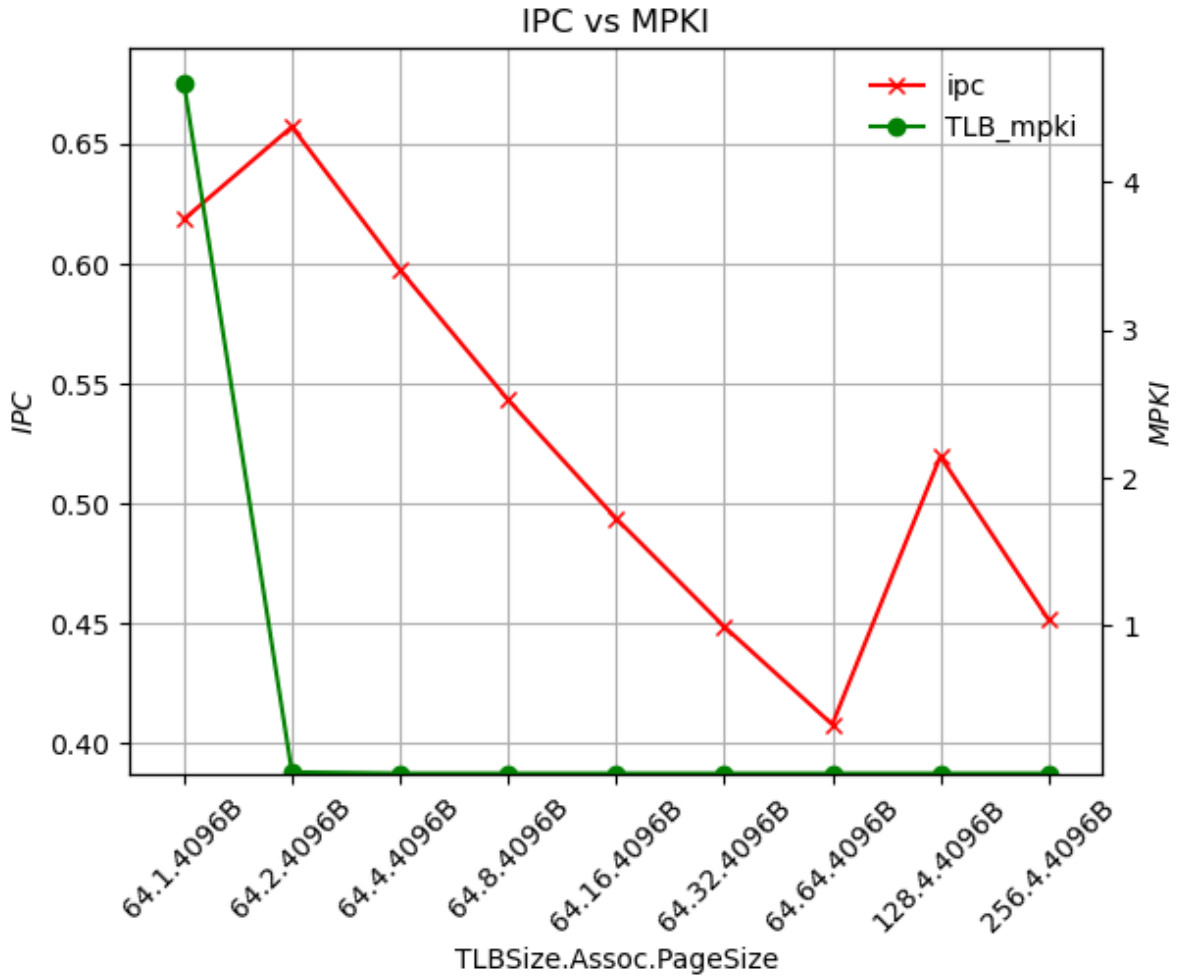
2.3.6 rtview (clock cycle architecture-dependent)



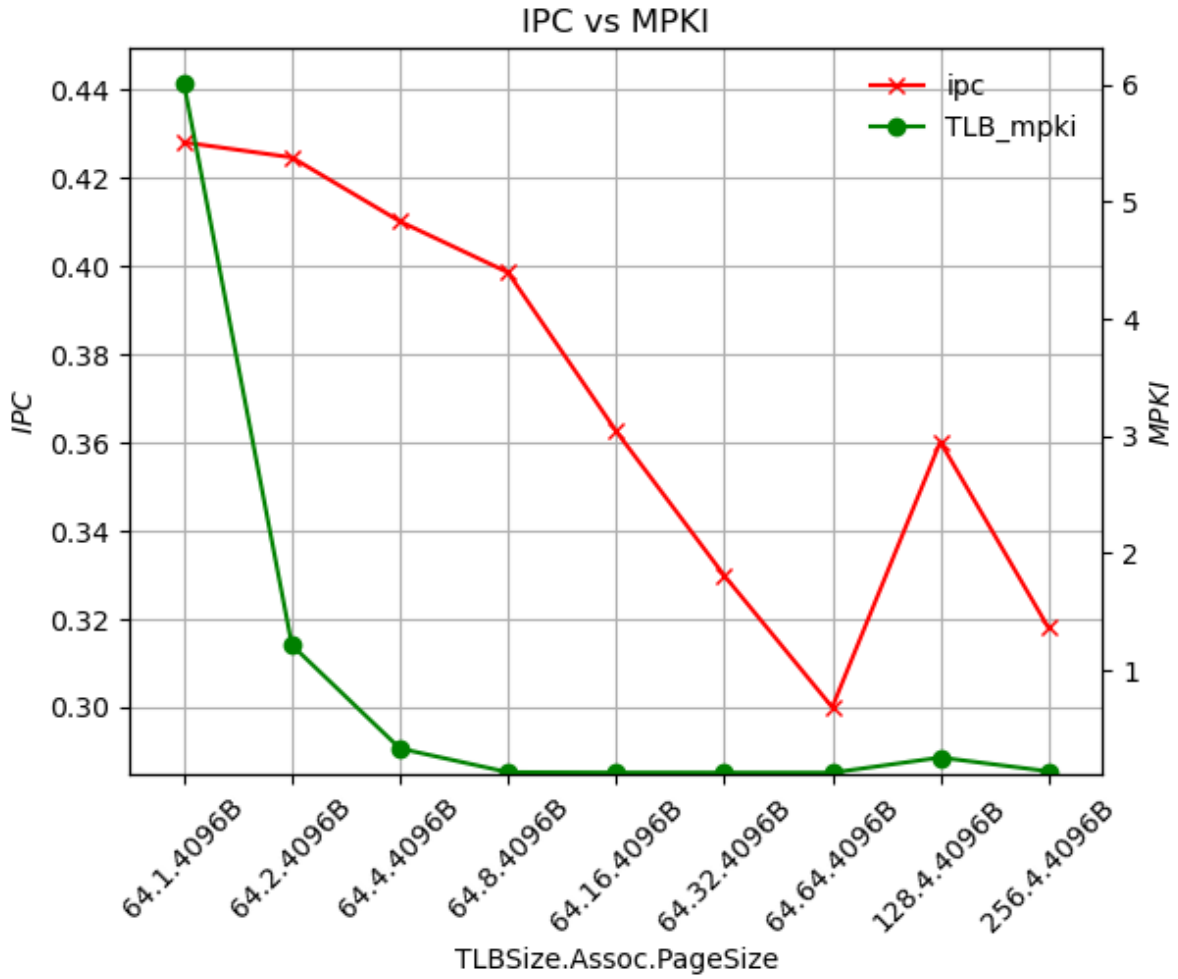
2.3.7 streamcluster (clock cycle architecture-dependent)



2.3.8 swaptions (clock cycle architecture-dependent)



2.3.9 Γεωμετρικός μέσος των benchmarks



2.3.10 Γενικές παρατηρήσεις για το TLB (κύκλος ρολογιού μεταβλητός)

- Καλύτερη επιλογή φαίνεται να είναι η τριπλέτα:

  `(TLB entries, associativity, page size) = (64, 1, 4096B)`

- Παρατηρείται η ίδια πτωτική τάση με πριν. Ούτε το TLB size ούτε το associativity επωφελούν κάπως το IPC όταν αυξάνονται.