

3

ΠΕΡΙΟΡΙΣΜΟΙ ΤΟΥ ΠΑΡΑΛΛΗΛΙΣΜΟΥ ΕΠΙΠΕΔΟΥ ΕΝΤΟΛΩΝ

Οι επεξεργαστές αναπτύσσονται έχοντας τη δυνατότητα να εκτελούν πάρα πολλές παράλληλες λειτουργίες σε επίπεδο εντολών. . . .
Υπάρχει η προοπτική πολύ σημαντικών εξελίξεων στον παραλληλισμό επιπέδου εντολών.

J. Fisher

(1981), στο άρθρο που διαπρωτυπώθηκε πρώτη φορά ο όρος «παραλληλισμός επιπέδου εντολών»

Μία από τις εκπλήξεις όσον αφορά την IA-64 είναι το γεγονός ότι δεν γινόμαστε αποδέκτες ισχυρισμών για υψηλή συχνότητα, παρά το ότι ο επεξεργαστής EPIC θεωρείται πως είναι λιγότερο πολύπλοκος από τους υπερβαθμωτούς επεξεργαστές. Είναι δύσκολο να το κατανοήσουμε αυτό, ωστόσο μπορούμε να εικάσουμε ότι η συνολική πολυπλοκότητα που περιλαμβάνει η επικέντρωση της προσοχής στο CPI, όπως συμβαίνει στην περίπτωση της IA-64, καθιστά δύσκολη την επίτευξη υψηλής συχνότητας.

M. Hopkins

(2000), στο σχολιασμό της αρχιτεκτονικής IA-64, που ήταν αποτέλεσμα της συνεργασίας της HP και της Intel η οποία είχε ως στόχο την επίτευξη σημαντικής αύξησης στην αξιοποίηση του ILP, χρησιμοποιώντας απλή αρχιτεκτονική, η οποία θα επέτρεπε υψηλότερη απόδοση

3.1 Εισαγωγή

Όπως επισημάναμε στο τελευταίο κεφάλαιο, η αξιοποίηση του ILP αποτέλεσε το βασικό σημείο ενδιαφέροντος των σχεδιάσεων των επεξεργαστών για περίπου 20 έτη, από τα μέσα της δεκαετίας του 1980 και έπειτα. Τα πρώτα 15 έτη σημειώθηκε σημαντική πρόοδος με την εμφάνιση διαδοχικών ολοένα και πιο προηγμένων προσεγγίσεων διασωλήνωσης, πολλαπλής έκδοσης, δυναμικής χρονοδρομολόγησης και εικασίας. Από το 2000 και έπειτα, οι σχεδιαστές έχουν στρέψει κυρίως το ενδιαφέρον τους στη βελτιστοποίηση των σχεδιάσεων ή στην προσπάθεια επίτευξης υψηλότερου ρυθμού ρολογιού, χωρίς την αύξηση του ρυθμού έκδοσης. Στο τέλος του τελευταίου κεφαλαίου επισημάναμε ότι αυτή η εποχή της προόδου, όσον αφορά την αξιοποίηση του ILP, φαίνεται πως πλησιάζει στο τέλος της.

Στο κεφάλαιο αυτό, ξεκινούμε με την εξέταση των περιορισμών του ILP που τίθενται εξαιτίας της δομής του προγράμματος, των ρεαλιστικών προϋποθέσεων του προϋπολογισμού για το υλικό, καθώς και της ακρίβειας των σημαντικών τεχνικών εικασίας, όπως η πρόβλεψη διακλάδωσης. Στην Ενότητα 3.5, εξετάζουμε τη χρήση του παραλληλισμού επιπέδου νημάτων ως εναλλακτική λύση ή προσθήκη στον παραλληλισμό επιπέδου εντολών. Τέλος, το κεφάλαιο κλείνει με τη σύγκριση συνόλου πρόσφατων επεξεργαστών τόσο όσον αφορά την απόδοση όσο και αναφορικά με τις μετρήσεις αποδοτικότητας ανά transistor και ανά watt.

3.2 Μελέτες Σχετικές με τους Περιορισμούς του ILP

Η αξιοποίηση του ILP για την αύξηση της απόδοσης ξεκίνησε στη δεκαετία του 1960 με τους πρώτους επεξεργαστές διασωλήνωσης. Οι τεχνικές αυτές, στις δεκαετίες του 1980 και του 1990, διαδραμάτισαν πρωτεύοντα ρόλο για την επίτευξη ταχύτατων βελτιώσεων στην απόδοση. Το ερώτημα που αφορά τον υφιστάμενο βαθμό του ILP ήταν ιδιαίτερα σημαντικό για τη μακροπρόθεσμη ικανότητά μας να οδηγούμαστε σε βελτιώσεις με ρυθμό που υπερέβαινε την αύξηση της ταχύτητας της βασικής τεχνολογίας ολοκληρωμένων κυκλωμάτων. Σε μικρότερη κλίμακα, το σημαντικό ερώτημα που αφορά τις απαιτήσεις αξιοποίησης περισσότερο ILP είναι κρίσιμο τόσο για τους σχεδιαστές υπολογιστών όσο και για τους δημιουργούς των μεταγλωττιστών. Τα δεδομένα, που παρατίθενται στην ενότητα αυτή, μας παρέχουν τρόπο εξέτασης της αξίας των ιδεών που έχουμε εισαγάγει στο τελευταίο κεφάλαιο, συμπεριλαμβανομένης της αποσαφήνισης (memory disambiguation) της μνήμης, της μετονομασίας των καταχωρητών και της εικασίας.

Στην ενότητα αυτή εξετάζουμε μία από τις μελέτες που εκπονήθηκαν για αυτά τα ερωτήματα. Η ενότητα της ιστορικής προοπτικής του Παραρτήματος Κ περιγράφει αρκετές μελέτες, συμπεριλαμβανομένης και αυτής από την οποία

προέρχονται τα δεδομένα της ενότητας αυτής (της μελέτης του Wall το 1993). Όλες αυτές οι μελέτες, που εξετάζουν το διαθέσιμο παραλληλισμό, λειτουργούν θέτοντας σύνολο προϋποθέσεων και διαπιστώνοντας το εύρος του διαθέσιμου παραλληλισμού όταν αυτές ισχύουν. Στην πραγματικότητα, είναι ουτοπικό να περιμένει κανείς τη δημιουργία του απόλυτα ακριβούς μοντέλου. Παρ' όλα αυτά, όλες αυτές οι μελέτες προϋποθέτουν τη χρήση ενός δεδομένου επιπέδου τεχνολογίας μεταγλωττιστή, ενώ κάποιες από τις προϋποθέσεις μπορούν να επηρεάσουν τα αποτελέσματα, παρά τη χρήση ιδιαίτερα φιλόδοξων μηχανισμών υλικού.

Στο μέλλον, οι εξελίξεις στον τομέα της τεχνολογίας των μεταγλωττιστών σε συνδυασμό με τις νέες και διαφορετικές σε μεγάλο βαθμό τεχνικές υλικού μπορεί να είναι σε θέση να αντιπαρέλθουν μερικούς από τους περιορισμούς που αποτελούν προϋπόθεση των μελετών αυτών. Ωστόσο, οι εξελίξεις αυτές, όταν συνδυαστούν με ρεαλιστικές τεχνικές υλικού, είναι απίθανο να υπερκεράσουν τους περιορισμούς αυτούς στο κοντινό μέλλον. Για παράδειγμα, η πρόβλεψη τιμής, την οποία εξετάσαμε στο προηγούμενο κεφάλαιο, μπορεί να αφαιρέσει τους περιορισμούς της εξάρτησης δεδομένων. Παρ' όλα αυτά, προκειμένου η πρόβλεψη τιμής να έχει σημαντικό αντίκτυπο στην απόδοση, οι μηχανισμοί πρόβλεψης χρειάζεται να διαθέτουν πολύ υψηλότερη ακρίβεια από αυτήν που μέχρι στιγμής έχει επιτευχθεί. Πράγματι, εξαιτίας αιτιών που θα εξετάσουμε στην Ενότητα 3.6, πιθανότατα έχουμε προσεγγίσει τα όρια του εύρους του ILP που μπορεί να αξιοποιηθεί με αποδοτικό τρόπο. Η ενότητα αυτή αποτελεί το κατάλληλο υπόβαθρο για την κατανόηση αυτής της περίπτωσης.

Το Μοντέλο Υλικού

Για να διαπιστώσουμε ποιοι είναι οι περιορισμοί του ILP πρέπει πρώτα να ορίσουμε τον ιδανικό επεξεργαστή. Ο ιδανικός επεξεργαστής είναι αυτός που δεν διαθέτει κανέναν από τους περιορισμούς του ILP. Στον επεξεργαστή αυτό, οι μόνοι περιορισμοί που υφίστανται είναι αυτοί που τίθενται από την πραγματική ροή των δεδομένων, είτε μέσω των καταχωρητών είτε μέσω της μνήμης.

Οι προϋποθέσεις που πρέπει να ισχύουν για τον ιδανικό επεξεργαστή είναι οι ακόλουθες:

1. *Μειωτομασία καταχωρητών*: Υπάρχει απεριόριστος αριθμός διαθέσιμων εικονικών καταχωρητών και ως εκ τούτου όλοι οι κίνδυνοι WAW και WAR αποφεύγονται, ενώ συνάμα υπάρχει απεριόριστος αριθμός εντολών που μπορούν να ξεκινούν να εκτελούνται ταυτόχρονα.
2. *Πρόβλεψη διακλάδωσης*: Η πρόβλεψη διακλάδωσης είναι απόλυτα ακριβής (perfect). Η πρόβλεψη όλων των διακλαδώσεων υπό συνθήκη πραγματοποιείται με αρτιότητα.
3. *Πρόβλεψη άψματος*: Υπάρχει απόλυτη ακρίβεια πρόβλεψης όλων των αλ-

μάτων (συμπεριλαμβανομένης και της εντολής καταχωρητή αλμάτων που χρησιμοποιείται για τα άλματα επιστροφής και τα υπολογισμένα άλματα). Ο συνδυασμός του παραπάνω με την απόλυτα ακριβή πρόβλεψη διακλάδωσης είναι ανάλογος ενός επεξεργαστή που διαθέτει απόλυτα ακριβή εικασία και απεριόριστου μεγέθους απομονωτή διαθέσιμων προς εκτέλεση εντολών.

4. *Ανάλυση ψευδωνυμίας διευθύνσεων μνήμης*: Όλες οι διευθύνσεις της μνήμης είναι επακριβώς γνωστές, ενώ μια εντολή φόρτωσης μπορεί να μεταφερθεί πριν από κάποια εντολή αποθήκευσης με την προϋπόθεση ότι δεν χρησιμοποιούν τις ίδιες διευθύνσεις. Αξίζει να σημειωθεί ότι αυτό οδηγεί σε απόλυτα ακριβή ανάλυση ψευδωνυμίας διευθύνσεων.
5. *Απόλυτα ακριβείς κρυφές μνήμες*: Όλες οι προσπελάσεις της μνήμης χρειάζονται 1 κύκλο ρολογιού. Στην πράξη, οι υπερβαθμωτοί επεξεργαστές συνήθως καταναλώνουν μεγάλο εύρος του ILP για την απόκρυψη των αστοχιών της κρυφής μνήμης, καθιστώντας τον παραπάνω στόχο ιδιαίτερα φιλόδοξο.

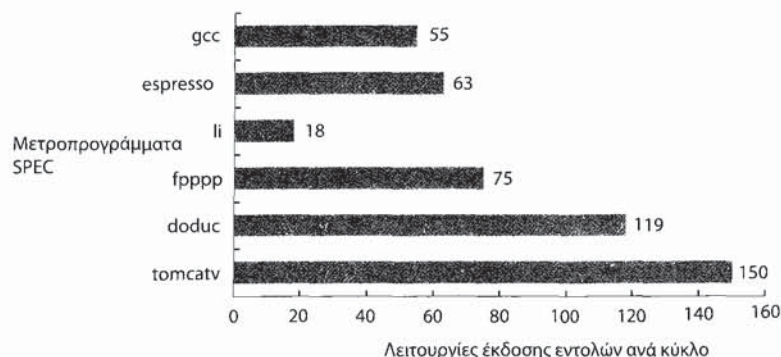
Οι προϋποθέσεις 2 και 3 εξαλείφουν *όλες* τις εξαρτήσεις ελέγχου. Παρόμοια, οι προϋποθέσεις 1 και 4 εξαλείφουν *όλες* τις εξαρτήσεις δεδομένων *εκτός από τις πραγματικές εξαρτήσεις δεδομένων*. Ο συνδυασμός των 4 αυτών προϋποθέσεων συνεπάγεται ότι είναι εφικτή η χρονοδρομολόγηση *οποιασδήποτε* εντολής της εκτέλεσης του προγράμματος στον αμέσως επόμενο κύκλο από αυτόν κατά τον οποίο πραγματοποιείται η εκτέλεση της εντολής από την οποία εξαρτάται. Υπό τις προϋποθέσεις αυτές, είναι εφικτό ακόμα και η *τελευταία* εντολή που εκτελείται με δυναμικό τρόπο να χρονοδρομολογηθεί στον πρώτο κύκλο! Συνεπώς, αυτό το σύνολο των προϋποθέσεων περιλαμβάνει τόσο τον έλεγχο όσο και την εικασία διευθύνσεων, διαδικασίες τις οποίες υλοποιεί με απόλυτη ακρίβεια.

Αρχικά, εξετάζουμε επεξεργαστή ο οποίος μπορεί να εκδίδει ταυτόχρονα απεριόριστο αριθμό εντολών αναζητώντας αυθαίρετα εντολές σε μεγάλη απόσταση εντός της ροής του υπολογισμού. Σε όλα τα μοντέλα επεξεργαστών που εξετάζουμε, υπάρχουν περιορισμοί όσον αφορά τους τύπους των εντολών που μπορούν να εκτελούν εντός ενός κύκλου. Στην περίπτωση του επεξεργαστή απεριόριστης έκδοσης, αυτό σημαίνει ότι μπορεί να υπάρχει απεριόριστος αριθμός εντολών φόρτωσης ή αποθήκευσης οι οποίες εκδίδονται εντός ενός κύκλου ρολογιού. Επιπλέον, όλοι οι λανθάνοντες χρόνοι των λειτουργικών μονάδων θεωρείται ότι ισούνται με 1 κύκλο, έτσι ώστε να είναι εφικτή η έκδοση οποιασδήποτε ακολουθίας εξαρτώμενων εντολών στον επόμενο κύκλο. Λανθάνοντες χρόνοι μεγαλύτεροι του ενός κύκλου θα οδηγούσαν στη μείωση του αριθμού της έκδοσης εντολών ανά κύκλο, αλλά όχι και του πλήθους των εντολών που θα εκτελούνταν ανά πάσα στιγμή (οι εντολές που ανά πάσα στιγμή βρίσκονται σε εκτέλεση συχνά αναφέρονται ως εντολές *εν πτήση*).

Βεβαίως, ο επεξεργαστής αυτός αγγίζει τα όρια της ουτοπίας. Για παράδειγμα, ο Power 5 της IBM είναι ένας από τους πλέον προηγμένους υπερβαθμωτούς επεξεργαστές που έχουν μέχρι σήμερα τεθεί σε κυκλοφορία. Ο Power 5 εκδίδει έως και τέσσερις εντολές ανά κύκλο και ξεκινά την εκτέλεση έως και έξι εντολών (με σημαντικούς περιορισμούς σε ό,τι αφορά τον τύπο των εντολών, π.χ. το πολύ δύο εντολές φόρτωσης - αποθήκευσης), υποστηρίζει μεγάλο αριθμό καταχωρητών μετονομασίας (88 καταχωρητών ακεραίων και 88 κινητής υποδιαστολής, επιτρέποντας την ύπαρξη περισσότερων από 200 εντολών εν πτήση, 32 εκ των οποίων μπορούν να είναι εντολές φόρτωσης και 32 εντολές αποθήκευσης), χρησιμοποιεί μεγάλου μεγέθους εντατικό μηχανισμό πρόβλεψης διακλάδωσης και υποστηρίζει τη δυναμική αποσαφήνιση της μνήμης. Μετά την εξέταση του διαθέσιμου παραλληλισμού στον ιδανικό επεξεργαστή, θα διερευνήσουμε τον αντίκτυπο που έχει ο περιορισμός διάφορων χαρακτηριστικών.

Για τη μέτρηση του διαθέσιμου παραλληλισμού, σύνολο προγραμμάτων μεταγλωττίστηκε και βελτιστοποιήθηκε μέσω των τυποποιημένων μεταγλωττιστών βελτιστοποίησης της MIPS. Τα προγράμματα ρυθμίστηκαν και εκτελέστηκαν με τέτοιο τρόπο ώστε να παραγάγουν ένα ίχνος με αναφορές των εντολών και των δεδομένων. Στη συνέχεια, κάθε εντολή του ίχνους χρονοδρομολογήθηκε όσο το δυνατόν νωρίτερα, περιοριζόμενη μονάχα από τις εξαρτήσεις δεδομένων. Καθώς χρησιμοποιείται κάποιο ίχνος, η απόλυτα ακριβής πρόβλεψη διακλάδωσης και η απόλυτα ακριβής ανάλυση ψευδωνυμίας είναι εύκολο να πραγματοποιηθεί. Με τους μηχανισμούς αυτούς, οι εντολές μπορεί να χρονοδρομολογούνται πολύ νωρίτερα απ' ό,τι θα συνέβαινε σε διαφορετική περίπτωση, μετακινούμενες πριν από μεγάλο αριθμό εντολών από τις οποίες δεν έχουν εξάρτηση δεδομένων, συμπεριλαμβανομένων και των διακλαδώσεων, καθώς η πρόβλεψη των διακλαδώσεων πραγματοποιείται με απόλυτη ακρίβεια.

Το Σχήμα 3.1 παρουσιάζει το μέσο εύρος του διαθέσιμου παραλληλισμού για έξι από τα μετροπρογράμματα του SPEC92. Σε όλη αυτήν την ενότητα, ο παραλληλισμός υπολογίζεται με βάση το μέσο ρυθμό έκδοσης εντολών. Αξίζει να έχουμε κατά νου ότι όλες οι εντολές έχουν λανθάνοντα χρόνο του ενός κύκλου. Οι υψηλότεροι λανθάνοντες χρόνοι θα οδηγούσαν στη μείωση του μέσου αριθμού εντολών ανά κύκλο. Τρία από αυτά τα μετροπρογράμματα είναι εντατικά προγράμματα κινητής υποδιαστολής (τα `frpprr`, `doeduc` και `tomcatv`), ενώ τα υπόλοιπα τρία είναι προγράμματα ακεραίων. Δύο από τα μετροπρογράμματα κινητής υποδιαστολής (τα `frpprr` και `tomcatv`) διαθέτουν εκτεταμένο εύρος παραλληλισμού, ο οποίος θα μπορούσε να αξιοποιηθεί από κάποιον διανυσματικό επεξεργαστή ή κάποιον πολυεπεξεργαστή (ωστόσο η δομή του `frpprr` είναι σε κάποιο βαθμό ανοργάνωτη, καθώς έχουν πραγματοποιηθεί μερικοί χειροκίνητοι μετασχηματισμοί του κώδικα). Το πρόγραμμα `doeduc` διαθέτει εκτεταμένο εύρος παραλληλισμού, ωστόσο ο παραλληλισμός αυτός δεν σημειώνεται σε απλούς παράλληλους βρόχους, όπως συμβαίνει στην περίπτωση των `frpprr` και `tomcatv`. Το πρόγραμμα `li` είναι διερμηνέας (*interpreter*) της LISP, ο οποίος



Σχήμα 3.1 Ο διαθέσιμος ILP ενός απόλυτα ακριβούς επεξεργαστή για έξι από τα μετροπρογράμματα του SPEC92. Τα τρία πρώτα προγράμματα είναι προγράμματα ακεραίων, ενώ τα τελευταία τρία είναι προγράμματα κινητής υποδιαστολής. Τα προγράμματα κινητής υποδιαστολής χρησιμοποιούν εντατικά βρόχους και έχουν υψηλό εύρος παραλληλισμού επιπέδου βρόχου.

διαθέτει πολλές μικρές εξαρτήσεις.

Στις επόμενες ενότητες, περιορίζουμε τα χαρακτηριστικά του επεξεργαστή αυτού, με απώτερο στόχο να καταδείξουμε την επίδραση που έχουν οι διάφορες προϋποθέσεις, προτού καταλήξουμε στην εξέταση μερικών φιλόδοξων, ωστόσο, ρεαλιστικών επεξεργασιών.

Περιορισμοί του Μεγέθους του Παραθύρου και του Μέγιστου Αριθμού Έκδοσης

Η δημιουργία επεξεργασιών που πλησιάζουν την απόλυτα ακριβή πρόβλεψη διακλάδωσης και την απόλυτα ακριβή ανάλυση ψευδωνυμίας προϋποθέτει εκτεταμένη δυναμική ανάλυση, καθώς οι στατικές προσεγγίσεις κατά το χρόνο μεταγλώττισης δεν μπορούν να είναι απόλυτα ακριβείς. Βεβαίως, οι περισσότερες ρεαλιστικές δυναμικές προσεγγίσεις δεν πρόκειται να είναι απόλυτα ακριβείς, ωστόσο η χρήση δυναμικών προσεγγίσεων παρέχει τη δυνατότητα αποκάλυψης του παραλληλισμού που δεν είναι δυνατό να γίνει αντιληπτός από τη στατική ανάλυση κατά το χρόνο μεταγλώττισης. Συνεπώς, οι δυναμικοί επεξεργαστές μπορεί να πλησιάζουν περισσότερο το εύρος του παραλληλισμού που αποκαλύπτεται από τον ιδανικό μας επεξεργαστή.

Σε ποιο βαθμό μπορούν να πλησιάσουν οι πραγματικοί δυναμικά χρονοδρομολογημένοι υποθετικοί επεξεργαστές τον ιδανικό επεξεργαστή; Προκειμένου να απαντήσουμε στο ερώτημα αυτό, ας εξετάσουμε τι πρέπει να εκτελεί ο απόλυτα ακριβής επεξεργαστής:

1. Να αναζητά αυθαίρετα, εντός μεγάλης απόστασης της ροής εντολών, σύνολο εντολών προς έκδοση, προβλέποντας όλες τις διακλαδώσεις με απόλυτη

ακρίβεια.

2. Να μετονομάζει όλες τις χρήσεις των καταχωρητών, έτσι ώστε να αποφεύγονται οι κίνδυνοι WAR και WAW.
3. Να προσδιορίζει αν υφίστανται εξαρτήσεις δεδομένων μεταξύ των εντολών του πακέτου έκδοσης. Αν όντως υφίστανται, τότε πρέπει να προβαίνει στις απαραίτητες μετονομασίες.
4. Να προσδιορίζει αν υφίστανται εξαρτήσεις μνήμης μεταξύ των εντολών που εκδίδονται και να τις αντιμετωπίζει κατάλληλα.
5. Να παρέχει αρκετές αντιγραμμένες λειτουργικές μονάδες, έτσι ώστε να επιτρέπει την έκδοση των έτοιμων εντολών.

Όπως είναι φανερό, η ανάλυση αυτή είναι αρκετά πολύπλοκη. Για παράδειγμα, ο προσδιορισμός του αν υφίστανται εξαρτήσεις καταχωρητών ανάμεσα στις n εντολές που εκδίδονται, με την προϋπόθεση ότι όλες οι εντολές είναι εντολές καταχωρητή - καταχωρητή και ότι ο συνολικός αριθμός των καταχωρητών είναι απειρίσιτος απαιτεί

$$2n - 2 + 2n - 4 + \dots + 2 = 2 \sum_{i=1}^{n-1} i = 2 \frac{(n-1)n}{2} = n^2 - n$$

συγκρίσεις. Επομένως, η ανίχνευση των εξαρτήσεων ανάμεσα στις επόμενες 2000 εντολές - αποτελεί το προεπιλεγμένο μέγεθος που χρησιμοποιούμε σε αρκετά σχήματα - απαιτεί σχεδόν 4 εκατομμύρια συγκρίσεις! Ακόμα και η έκδοση 50 μόνο εντολών απαιτεί 2450 συγκρίσεις. Το κόστος αυτό, όπως είναι φανερό, περιορίζει τον αριθμό των εντολών που μπορεί να θεωρηθεί ότι θα εκδοθούν ταυτόχρονα.

Στους υφιστάμενους επεξεργαστές και σε αυτούς που πρόκειται να κυκλοφορήσουν σύντομα, τα κόστη δεν είναι τόσο υψηλά, καθώς χρειάζεται να ανιχνεύσουμε μονάχα ζεύγη εξαρτήσεων, ενώ ο περιορισμένος αριθμός των καταχωρητών επιτρέπει διαφορετικές λύσεις. Επιπλέον, στους πραγματικούς επεξεργαστές, η έκδοση πραγματοποιείται εντός σειράς, ενώ ο χειρισμός των εξαρτώμενων εντολών πραγματοποιείται με τη βοήθεια της διαδικασίας μετονομασίας η οποία εκτελεί τη μετονομασία των εξαρτώμενων εντολών εντός ενός κύκλου. Μόλις εκδοθούν οι εντολές, η ανίχνευση των εντολών υλοποιείται με κατανεμημένο τρόπο μέσω των σταθμών κράτησης ή του πίνακα παρακολούθησης.

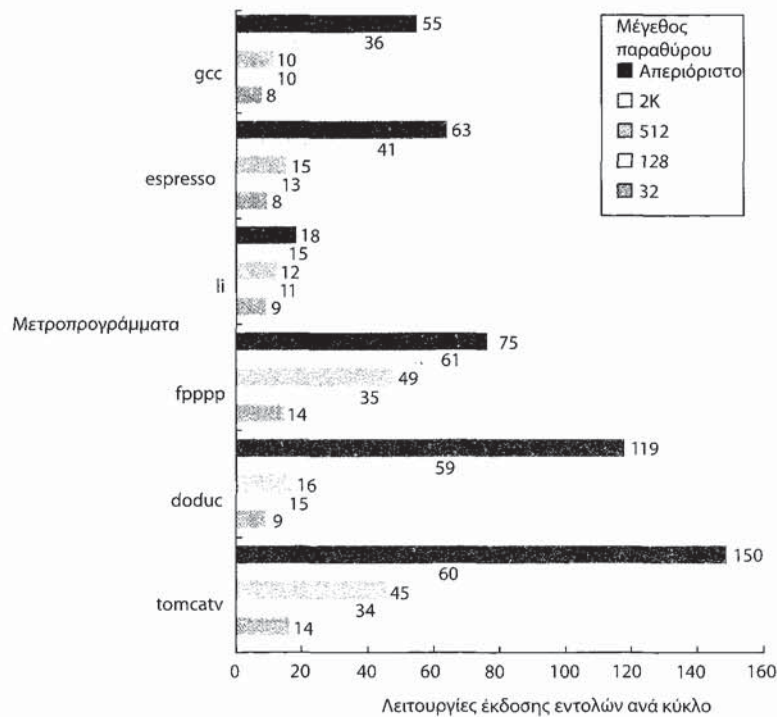
Το σύνολο των εντολών που εξετάζεται ως προς την ταυτόχρονη εκτέλεση ονομάζεται *παράθυρο* (window). Κάθε εντολή του παραθύρου πρέπει να διατηρείται στον επεξεργαστή, ενώ το πλήθος των συγκρίσεων που απαιτούνται σε κάθε κύκλο ρολογιού ισούται με το γινόμενο του μέγιστου ρυθμού ολοκλήρωσης με το

μέγεθος του παραθύρου επί τον αριθμό των ορισμάτων ανά εντολή (σήμερα έως και $6 \times 200 \times 2 = 2400$), καθώς κάθε εκκρεμής εντολή πρέπει να εξετάσει και τα δύο ορίσματα κάθε εντολής που ολοκληρώνεται. Επομένως, το συνολικό μέγεθος του παραθύρου περιορίζεται από την απαιτούμενη αποθήκευση, από τις συγκρίσεις, καθώς και από τον περιορισμένο ρυθμό έκδοσης, γεγονός που μειώνει τη χρησιμότητα των παραθύρων όσο αυξάνεται το μέγεθός τους. Ας έχουμε καιά νου ότι παρόλο που οι υφιστάμενοι επεξεργαστές επιτρέπουν την ύπαρξη εκατοντάδων εν πτήσει εντολών, το γεγονός ότι μπορούν να εκδίδουν και να μετονομάζουν πολύ μικρό αριθμό εντολών ανά κύκλο ρολογιού οδηγεί στον πιθανό περιορισμό της ρυθμοαπόδοσης εξαιτίας του ρυθμού έκδοσης. Για παράδειγμα, αν η ροή εντολών περιελάμβανε εντελώς ανεξάρτητες εντολές, για όλες εκ των οποίων θα σημειωνόταν ευστοχία της κρυφής μνήμης (cache hit), τότε δεν θα οδηγούμασταν απλά ποτέ στην εξ ολοκλήρου συμπλήρωση ενός μεγάλου παραθύρου. Η χρησιμοποίηση παραθύρου με μεγαλύτερο μέγεθος από το ρυθμό έκδοσης είναι πολύτιμη σε περιπτώσεις όπου είτε υπάρχουν εξαρτήσεις εντός της ροής των εντολών είτε σημειώνονται αστοχίες στην κρυφή μνήμη.

Το μέγεθος του παραθύρου περιορίζει με άμεσο τρόπο τον αριθμό των εντολών που ξεκινούν να εκτελούνται ενός κάποιου δεδομένου κύκλου. Στην πράξη, οι πραγματικοί επεξεργαστές διαθέτουν ακόμα πιο περιορισμένο αριθμό λειτουργικών μονάδων (π.χ. κανένας υπερβαθμωτός επεξεργαστής δεν χειρίζεται περισσότερες από δύο αναφορές στη μνήμη ανά κύκλο ρολογιού), καθώς και περιορισμένο πλήθος διαύλων και θυρών πρόσβασης στους καταχωρητές, οι οποίοι θέτουν το όριο των εντολών που μπορούν να αρχικοποιηθούν ανά κύκλο ρολογιού. Συνεπώς, ο μέγιστος αριθμός εντολών που μπορούν να εκδοθούν, να ξεκινήσουν να εκτελούνται ή να επικυρωθούν εντός του ίδιου κύκλου ρολογιού είναι συνήθως πολύ μικρότερος από το μέγεθος του παραθύρου.

Όπως είναι φανερό, το πλήθος των ενδεχόμενων περιορισμών υλοποίησης στους επεξεργαστές πολλαπλής έκδοσης είναι μεγάλο, περιορισμών που αφορούν τις εντολές που εκδίδονται ανά κύκλο, τις λειτουργικές μονάδες και το λανθάνοντα χρόνο των μονάδων, τις θύρες της συστοιχίας των καταχωρητών, τις ουρές των λειτουργικών μονάδων (οι οποίες μπορεί να είναι λιγότερες από τις μονάδες), την έκδοση των διακλαδώσεων και την επικύρωση των εντολών. Καθένα από τα παραπάνω παραδείγματα ενεργεί ως περιορισμός του ILP. Ωστόσο, αντί να επιχειρήσουμε να κατανοήσουμε τις επιπτώσεις που έχει καθένας από τους παραπάνω περιορισμούς, θα στρέψουμε το ενδιαφέρον μας στον περιορισμό του μεγέθους του παραθύρου, έχοντας κατά νου ότι όλοι οι υπόλοιποι περιορισμοί οδηγούν στην περαιτέρω μείωση του εύρους του παραλληλισμού που μπορεί να αξιοποιηθεί.

Το Σχήμα 3.2 παρουσιάζει τις επιπτώσεις που προκύπτουν από τον περιορισμό του μεγέθους του παραθύρου από το οποίο μπορούν να εκτελούνται οι εντολές. Όπως μπορεί να διαπιστώσει κανείς παρατηρώντας το Σχήμα 3.2, το εύρος του παραλληλισμού μειώνεται δραστικά παράλληλα με τη μείωση του



Σχήμα 3.2 Η επίδραση του μεγέθους του παραθύρου σε κάθε εφαρμογή, παρουσιάζοντας στο διάγραμμα το μέσο αριθμό έκδοσης εντολών ανά κύκλο ρολογιού.

μεγέθους του παραθύρου. Το 2005, οι πιο προηγμένοι επεξεργαστές διαθέτουν μέγεθος παραθύρου που κυμαίνονται ανάμεσα στις 64 και τις 200 εντολές, ωστόσο το μέγεθος αυτό δεν μπορεί να συγκριθεί με αυστηρό τρόπο με τα μεγέθη που χρησιμοποιούνται στο Σχήμα 3.2 για δύο λόγους. Πρώτον, πολλές λειτουργικές μονάδες διαθέτουν λανθάνοντα χρόνο πολλαπλών κύκλων, κάτι που οδηγεί στη μείωση του μεγέθους ενεργού παραθύρου σε σύγκριση με την περίπτωση κατά την οποία όλες οι μονάδες διαθέτουν λανθάνοντα χρόνο του ενός κύκλου. Δεύτερον, στους πραγματικούς επεξεργαστές, στο παράθυρο πρέπει να διατηρούνται, επίσης, όλες οι αναφορές στη μνήμη που βρίσκονται σε αναμονή εξαιτίας της αστοχίας της κρυφής μνήμης, κάτι το οποίο δεν εξετάζεται στο μοντέλο αυτό, καθώς θεωρείται ότι πραγματοποιείται απόλυτα ακριβής προσπέλαση της κρυφής μνήμης εντός ενός κύκλου.

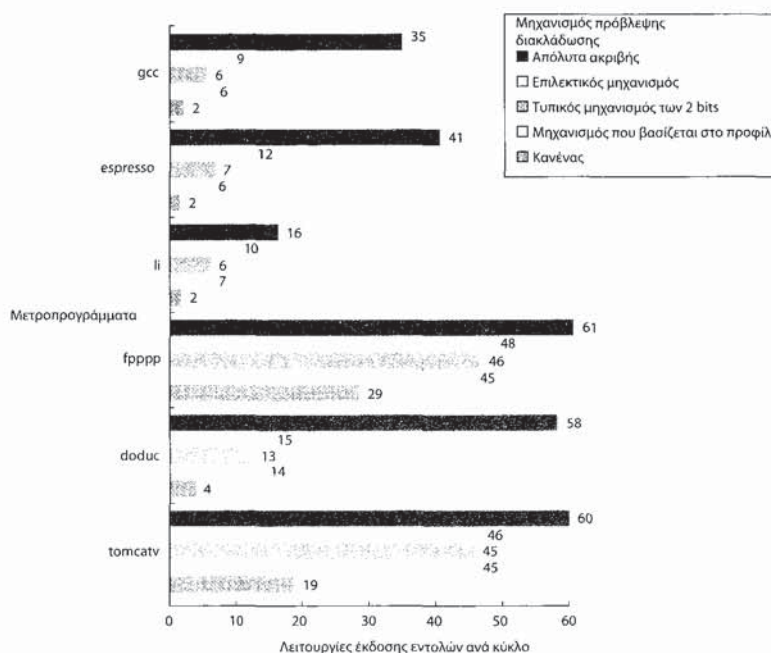
Όπως μπορούμε να διακρίνουμε στο Σχήμα 3.2, τα προγράμματα ακεραίων δεν διαθέτουν το ίδιο εύρος παραλληλισμού με τα προγράμματα κινητής υποδιαστολής. Το αποτέλεσμα αυτό είναι αναμενόμενο. Παρακολουθώντας τον τρόπο με τον οποίο μειώνεται ο παραλληλισμός στο Σχήμα 3.2 είναι εύκολο να κατανοήσουμε ότι στις περιπτώσεις της κινητής υποδιαστολής ο παραλληλισμός έχει τις ρίζες του στον παραλληλισμό επιπέδου βρόχου. Το γεγονός ότι το εύ-

ρος του παραλληλισμού των περιπτώσεων όπου χρησιμοποιείται μικρό μέγεθος παραθύρου δεν έχει μεγάλες διαφορές μεταξύ των προγραμμάτων κινητής υποδιαστολής και ακεραίων υποδηλώνει την ύπαρξη δομής στην οποία υπάρχουν εξαρτήσεις εντός του σώματος των βρόχων και λιγότερες εξαρτήσεις μεταξύ των επαναλήψεων του βρόχου, κάτι που είναι ιδιαίτερα ευδιάκριτο σε προγράμματα όπως το `tomcatv`. Ο επεξεργαστής, όταν χρησιμοποιεί μικρό μέγεθος παραθύρου, απλά δεν μπορεί να έχει πρόσβαση στις εντολές της επόμενης επανάληψης του βρόχου οι οποίες θα μπορούσαν να εκδοθούν παράλληλα με τις εντολές της τρέχουσας επανάληψης. Η κατάσταση αυτή αποτελεί ενδεικτικό παράδειγμα των περιπτώσεων στις οποίες η τεχνολογία των μεταγωγτιστών (δείτε το Παράρτημα Z) μπορεί να αποκαλύψει υψηλότερο εύρος ILP, καθώς είναι σε θέση να εντοπίζει τον παραλληλισμό επιπέδου βρόχου και να χρονοδρομολογεί τον κώδικα με τέτοιο τρόπο, ώστε να αξιοποιεί το γεγονός αυτό, ακόμα και όταν το μέγεθος παραθύρου είναι μικρό.

Γνωρίζουμε ότι τα πολύ μεγάλα μεγέθη παραθύρου δεν είναι πρακτικά και αποδοτικά και το Σχήμα 3.2 μας ενημερώνει ότι η ρυθμοαπόδοση των εντολών είναι ιδιαίτερα μειωμένη στις πραγματικές υλοποιήσεις. Συνεπώς, στην υπόλοιπη ανάλυσή μας, θα χρησιμοποιούμε ως βάση εκκίνησης μεγέθη παραθύρων της τάξης των 2K καταχωρίσεων, τα οποία είναι σχεδόν 10 φορές μεγαλύτερα από το μέγεθος που διέθετε η μεγαλύτερη υλοποίηση κατά το έτος 2005, ενώ ο μέγιστος ρυθμός έκδοσης ισούται με 64 εντολές ανά κύκλο ρολογιού, ο οποίος, επίσης, είναι 10 φορές μεγαλύτερο από τον υψηλότερο ρυθμό που σημείωνε κάποιος επεξεργαστής κατά το 2005. Όπως θα διαπιστώσουμε στις επόμενες ενότητες, όταν τα υπόλοιπα τμήματα του επεξεργαστή δεν είναι απόλυτα ακριβή, τα παράθυρα με μέγεθος 2K σε συνδυασμό με τον περιορισμό της έκδοσης 64 εντολών δεν περιορίζουν το εύρος του ILP που μπορεί να αξιοποιηθεί ο επεξεργαστής.

Η Επίδραση της Ρεαλιστικής Πρόβλεψης Διακλαδώσεων και Αλμάτων

Στον ιδανικό μας επεξεργαστή θεωρούμε ότι η πρόβλεψη διακλάδωσης είναι απόλυτα ακριβής: Το αποτέλεσμα κάθε διακλάδωσης του προγράμματος είναι γνωστό πριν την εκτέλεση της πρώτης εντολής! Βεβαίως, κανένας πραγματικός επεξεργαστής δεν μπορεί να το επιτύχει αυτό. Το Σχήμα 3.3 παρουσιάζει την επίδραση των πιο ρεαλιστικών προσεγγίσεων πρόβλεψης σε δύο διαφορετικές μορφές. Τα δεδομένα μας αφορούν πολλές διαφορετικές προσεγγίσεις πρόβλεψης διακλάδωσης, από αυτές που είναι απόλυτα ακριβείς έως και αυτές που δεν χρησιμοποιούν κάποιο μηχανισμό διακλάδωσης. Θεωρούμε ότι για τα άλματα χρησιμοποιείται ξεχωριστός μηχανισμός πρόβλεψης. Οι μηχανισμοί πρόβλεψης αλμάτων διαδραματίζουν σημαντικό ρόλο κυρίως στις περιπτώσεις όπου υπάρχουν οι πιο ακριβείς μηχανισμοί πρόβλεψης διακλάδωσης, καθώς η συχνότητα των διακλαδώσεων είναι υψηλότερη και η ακρίβεια των μηχανισμών πρόβλεψης



Σχήμα 3.3 Η επίδραση των προσεγγίσεων πρόβλεψης διακλάδωσης ταξινομημένες ως προς κάθε εφαρμογή. Το γράφημα αυτό παρουσιάζει την επίδραση που έχει η υιοθέτηση ενός απόλυτα ακριβούς μοντέλου πρόβλεψης διακλάδωσης (όπου πραγματοποιείται ορθή πρόβλεψη όλων των εντολών ανεξαρτήτως της απόστασής τους από τη ροή των εντολών), η χρήση διάφορων δυναμικών μηχανισμών πρόβλεψης (επιλεκτικών μηχανισμών και μηχανισμών των 2 bits) και, τέλος, η μη χρησιμοποίηση κανενός μηχανισμού πρόβλεψης. Οι μηχανισμοί πρόβλεψης περιγράφονται λεπτομερώς στο κείμενο του βιβλίου. Το γράφημα αυτό επισημαίνει τις διαφορές ανάμεσα στα προγράμματα που διαθέτουν εκτεταμένο παραλληλισμό επιπέδου βρόχου (στο tomcatv και το frrpp) και σε αυτά που δεν διαθέτουν αυτή την ιδιότητα (στα προγράμματα ακεραίων και στο doduc).

διακλάδωσης είναι ζήτημα κεφαλαιώδους σημασίας.

Τα πέντε επίπεδα της πρόβλεψης διακλάδωσης που παρουσιάζονται στο σχήμα αυτό είναι τα ακόλουθα:

1. *Απόλυτη ακρίβεια:* Όλες οι διακλαδώσεις και τα άλματα προβλέπονται με απόλυτη ακρίβεια από την έναρξη της εκτέλεσης.
2. *Επιλεκτικοί μηχανισμοί πρόβλεψης διακλάδωσης:* Αυτή η προσέγγιση πρόβλεψης χρησιμοποιεί ένα συσχετικό μηχανισμό πρόβλεψης διακλάδωσης των 2 bits και ένα μη συσχετικό μηχανισμό διακλάδωσης των 2 bits σε συνδυασμό με ένα μηχανισμό επιλογής (selector), ο οποίος επιλέγει τον καλύτερο μηχανισμό πρόβλεψης για κάθε διακλάδωση. Ο απομονωτής πρόβλεψης περιλαμβάνει 2^{13} (8K) καταχωρίσεις, καθεμία εκ των οποίων

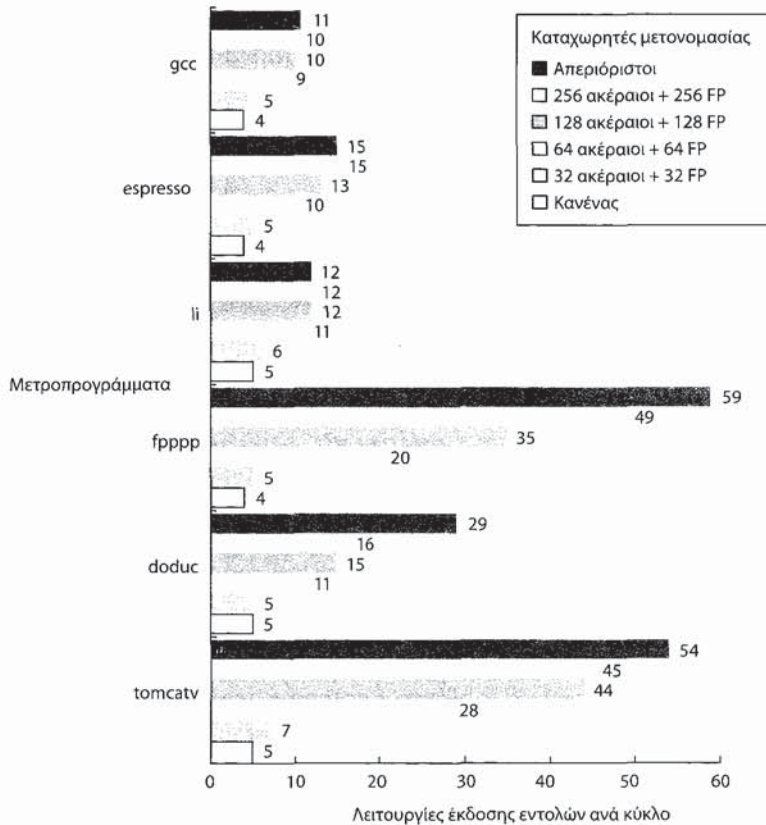
έκδοσης και του παραθύρου, θα θεωρούμε ως βάση σύγκρισης έναν πιο φιλόδοξο επιλεκτικό μηχανισμό πρόβλεψης που χρησιμοποιεί δύο επίπεδα πρόβλεψης και συνολικά 8K καταχωρίσεις. Αυτός ο μηχανισμός πρόβλεψης, ο οποίος απαιτεί περισσότερα από 150K bits αποθήκευσης (είναι σχεδόν τετραπλάσιο μέγεθος από το μεγαλύτερο μηχανισμό πρόβλεψης μέχρι σήμερα), παρουσιάζει ελαφρά υπεροχή σε σύγκριση με τον επιλεκτικό μηχανισμό πρόβλεψης που περιγράφθηκε παραπάνω (υπεροχή της τάξης του 0.5%-1%). Επίσης, θεωρούμε ότι χρησιμοποιείται ζεύγος μηχανισμών πρόβλεψης αλμάτων και επιστροφών των 2K, όπως περιγράφηκε παραπάνω.

Η Επίδραση των Πεπερασμένων Καταχωρητών

Ο ιδανικός μας επεξεργαστής εξαλείφει όλες τις εξαρτήσεις ονομάτων που υπάρχουν μεταξύ αναφορών στη μνήμη χρησιμοποιώντας απεριόριστο σύνολο εικονικών καταχωρητών. Σήμερα, ο Power 5 της IBM έχει παράσχει το μεγαλύτερο αριθμό εικονικών καταχωρητών: 88 επιπρόσθετων καταχωρητών κινητής υποδιαστολής και 88 επιπρόσθετων καταχωρητών ακεραίων σε συνδυασμό με τους 64 καταχωρητές που είναι διαθέσιμοι από τη βασική αρχιτεκτονική. Όταν πραγματοποιείται πολυνηματική εκτέλεση (δείτε την Ενότητα 3.5), δύο νήματα διαμοιράζονται και τους 240 καταχωρητές, ενώ, όταν πραγματοποιείται εκτέλεση ενός νήματος, όλοι οι καταχωρητές είναι διαθέσιμοι για το νήμα αυτό. Το Σχήμα 3.5 παρουσιάζει την επίδραση που έχει η μείωση του αριθμού των διαθέσιμων για μετονομασία καταχωρητών: το πλήθος τόσο των καταχωρητών FP όσο και των καταχωρητών GP¹ έχει αυξηθεί με βάση τον αριθμό των καταχωρητών που προσδιορίζονται στο υπόμνημα.

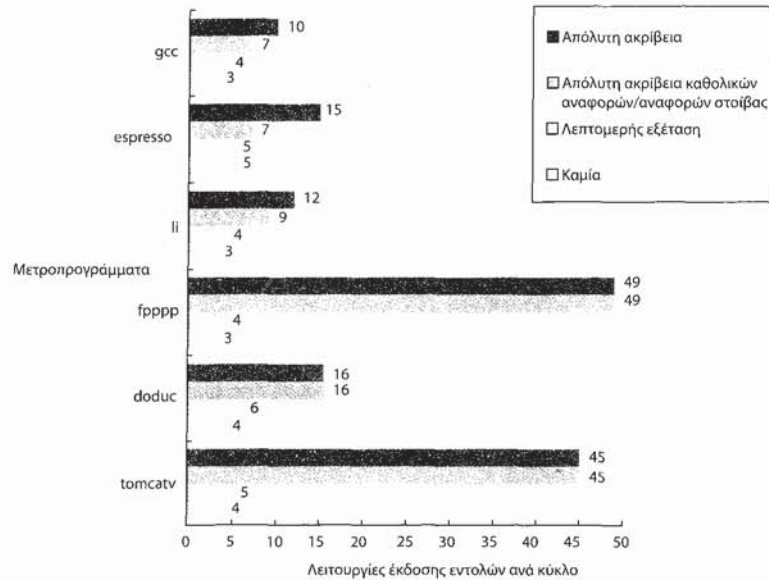
Τα αποτελέσματα του σχήματος αυτού μπορεί να προκαλούν σε κάποιο βαθμό έκπληξη: Θα ανέμενε κανείς ότι οι εξαρτήσεις ονομάτων θα έπρεπε να μειώνουν σε μικρό βαθμό το διαθέσιμο παραλληλισμό. Ωστόσο, αξίζει να θυμηθούμε ότι η αξιοποίηση του παραλληλισμού σε μεγάλο βαθμό απαιτεί την αξιολόγηση πολλών ενδεχόμενων διαδρομών εκτέλεσης με υποθετικό τρόπο. Συνεπώς, απαιτούνται πολλοί καταχωρητές για τη διατήρηση των μεταβλητών αυτών των νημάτων. Σύμφωνα με το Σχήμα 3.5 ο αντίκτυπος της χρήσης πεπερασμένου αριθμού καταχωρητών είναι σημαντικός, αν υφίσταται εκτεταμένος παραλληλισμός. Παρόλο που το γράφημα αυτό καταδεικνύει το μεγάλο αντίκτυπο που έχει η μείωση των καταχωρητών στα προγράμματα FP, η μείωση αυτή έχει χαμηλό αντίκτυπο στα προγράμματα ακεραίων κυρίως διότι οι περιορισμοί του μεγέθους του παραθύρου και της πρόβλεψης διακλάδωσης έχουν ήδη περιορίσει τον ILP σημαντικά, κάτι που οδηγεί στη μείωση της σπουδαιότητας της μετονομασίας. Επιπλέον, αξίζει να σημειωθεί ότι η μείωση του διαθέσιμου παραλληλισμού είναι σημαντική, ακόμα και αν υπάρχουν 64 επιπρόσθετοι καταχωρητές ακεραίων

¹Σ.τ.Μ. Ακρωνύμιο του αγγλικού όρου "general purpose" που στα ελληνικά μεταφράζεται ως «γενικού σκοπού»)



Σχήμα 3.5 Η μείωση του διαθέσιμου παραλληλισμού είναι σημαντική, όταν δεν υφίσταται απεριόριστος αριθμός διαθέσιμων καταχωρητών. Ο αριθμός τόσο των καταχωρητών FP όσο και των καταχωρητών GP έχει αυξηθεί με βάση τον αριθμό των καταχωρητών που προσδιορίζονται στο υπόμνημα. Επομένως, η περίπτωση που αντιστοιχεί στην επιλογή «128 ακέραιοι + 128 FP» διαθέτει συνολικά $128 + 128 + 64 = 320$ καταχωρητές (128 για τη μετονομασία των καταχωρητών ακεραίων, 128 για τη μετονομασία των καταχωρητών FP και 64 καταχωρητές ακεραίων και FP που χρησιμοποιεί η αρχιτεκτονική MIPS). Η επίδραση είναι ακόμα πιο σημαντική στα προγράμματα FP, παρόλο που η παρουσία μόνο 32 επιπλέον καταχωρητών ακεραίων και 32 επιπλέον καταχωρητών FP έχει σημαντικό αντίκτυπο σε όλα τα προγράμματα. Σε ό,τι αφορά τα προγράμματα ακεραίων, η παρουσία περισσότερων από 64 επιπλέον καταχωρητών δεν παρουσιάζεται στο σχήμα αυτό. Η χρήση περισσότερων από 64 καταχωρητών προϋποθέτει την αποκάλυψη του παραλληλισμού σε μεγάλο βαθμό, κάτι το οποίο, στην περίπτωση των προγραμμάτων ακεραίων, απαιτεί ουσιαστικά την απόλυτα ακριβή πρόβλεψη των διακλαδώσεων.

και 64 επιπρόσθετοι καταχωρητές FP διαθέσιμοι για μετονομασία, περίπτωση η οποία αντιστοιχεί στον αριθμό των επιπλέον καταχωρητών που υπάρχουν σε οποιονδήποτε επεξεργαστή του 2005.



Σχήμα 3.6 Η επίδραση των διάφορων επιπέδων ανάλυσης ψευδωνυμίας των μεμονωμένων προγραμμάτων. Οτιδήποτε άλλο πέρα από την απόλυτα ακριβή ανάλυση έχει ιδιαίτερα σημαντικές επιπτώσεις στο εύρος του διαθέσιμου παραλληλισμού στα προγράμματα ακεραίων, ενώ η ανάλυση καθολικών αναφορών/αναφορών στοίβας είναι απόλυτα ακριβής (και μη εφικτή) για τα προγράμματα της FORTRAN.

Παρόλο που η μετονομασία των καταχωρητών είναι εμφανώς σημαντική για την απόδοση, η χρήση απεριόριστου αριθμού καταχωρητών δεν είναι αποδοτική. Κατά συνέπεια, στην επόμενη ενότητα, θεωρούμε ότι υπάρχουν 256 καταχωρητές ακεραίων και 256 καταχωρητές FP διαθέσιμοι για μετονομασία - πολύ μεγαλύτερος αριθμός από οποιονδήποτε επεξεργαστή του 2005.

Η Επίδραση της Μη Απόλυτα Ακριβούς Ανάλυσης Ψευδωνυμίας

Στο βελτιστοποιημένο μοντέλο μας θεωρούμε ότι μπορούμε να αναλύσουμε όλες τις εξαρτήσεις της μνήμης με απόλυτη ακρίβεια και να εξαλείψουμε όλες τις εξαρτήσεις ονομάτων καταχωρητών. Βεβαίως, η απόλυτα ακριβής ανάλυση ψευδωνυμίας δεν είναι εφικτή στην πράξη: Η ανάλυση δεν μπορεί να είναι απόλυτα ακριβής κατά το χρόνο μεταγλώττισης, ενώ απαιτεί ενδεχομένως απεριόριστο αριθμό συγκρίσεων κατά το χρόνο εκτέλεσης (καθώς το πλήθος των ταυτόχρονων αναφορών στη μνήμη είναι απεριόριστο). Το Σχήμα 3.6 παρουσιάζει την επίδραση που έχουν τρία διαφορετικά μοντέλα ανάλυσης ψευδωνυμίας της μνήμης, καθώς και το μοντέλο της απόλυτα ακριβούς ανάλυσης. Τα τρία μοντέλα είναι τα ακόλουθα:

1. *Απόλυτη ακρίβεια πρόβλεψης καθολικών αναφορών/αναφορών στοίβας:* Το μοντέλο πραγματοποιεί απόλυτα ακριβείς προβλέψεις καθολικών αναφορών (global references) και αναφορών στοίβας (stack references), ενώ θεωρεί ότι όλες οι αναφορές του σωρού (heap references) έρχονται σε σύγκρουση. Το μοντέλο αυτό αποτελεί εξιδανικευμένη έκδοση των καλύτερων σύγχρονων προσεγγίσεων ανάλυσης που βασίζονται στο μεταγλωττιστή. Οι πρόσφατες εξελίξεις στο χώρο της έρευνας της ανάλυσης ψευδωνυμίας των δεικτών (pointers) πρόκειται να βελτιώσουν στο μέλλον το χειρισμό των δεικτών του σωρού.
2. *Λεπτομερής εξέταση:* Το μοντέλο αυτό εξετάζει τις προσπελάσεις για να διαπιστώσει, κατά το χρόνο μεταγλώττισης, αν αυτές επηρεάζονται μεταξύ τους. Για παράδειγμα, αν κάποια προσπέλαση χρησιμοποιεί τον R10 ως καταχωρητή βάσης με σχετική απόσταση ίση με 20, τότε κάποια άλλη προσπέλαση η οποία χρησιμοποιεί τον R10 ως καταχωρητή βάσης με σχετική απόσταση ίση με 100 δεν μπορεί να επηρεάσει την προηγούμενη προσπέλαση, με την προϋπόθεση ότι η τιμή του R10 δεν μπορεί να έχει αλλάξει. Επιπλέον, στις διευθύνσεις που βασίζονται σε καταχωρητές οι οποίοι παραπέμπουν σε διαφορετικές περιοχές δέσμευσης της μνήμης (όπως η καθολική περιοχή και η περιοχή στοίβας) θεωρείται ότι ποτέ δεν υφίσταται ψευδωνυμία. Η ανάλυση αυτή είναι παρόμοια με αυτήν που υλοποιείται από πολλούς υφιστάμενους εμπορικούς μεταγλωττιστές, παρόλο που οι νέοι μεταγλωττιστές διαθέτουν καλύτερη λειτουργικότητα, τουλάχιστον σε ό,τι αφορά τα προγράμματα με βρόχους.
3. *Καμία εξέταση:* Όλες οι αναφορές στη μνήμη θεωρείται ότι έρχονται σε σύγκρουση.

Όπως θα περίμενε κανείς, σε ό,τι αφορά τα προγράμματα της FORTRAN (στα οποία δεν υφίστανται αναφορές σωρού), δεν υπάρχει καμία διαφορά ανάμεσα στην απόλυτα ακριβή ανάλυση και στην απόλυτα ακριβή ανάλυση των καθολικών αναφορών/αναφορών στοίβας. Η απόλυτα ακριβής ανάλυση των καθολικών αναφορών/αναφορών στοίβας είναι βελτιστοποιημένη, καθώς δεν υπάρχει μεταγλωττιστής που θα μπορούσε να ανιχνεύσει επακριβώς όλες τις εξαρτήσεις πινάκων. Το γεγονός ότι η απόλυτα ακριβής ανάλυση των καθολικών αναφορών και των αναφορών στοίβας εξακολουθεί να είναι κατά δύο φορές καλύτερη από τη λεπτομερή εξέταση (inspection) καταδεικνύει ότι για την επίτευξη περισσότερου παραλληλισμού απαιτείται είτε προηγμένη ανάλυση σε επίπεδο μεταγλωττιστή είτε δυναμική ανάλυση κατά το χρόνο εκτέλεσης. Στην πράξη, οι δυναμικά χρονοδρομολογημένοι επεξεργαστές βασίζονται στη δυναμική αποσαφήνιση της μνήμης. Προκειμένου να υλοποιήσει κανείς την απόλυτα ακριβή δυναμική αποσαφήνιση κάποιας δεδομένης εντολής φόρτωσης, πρέπει να γνωρίζει τις διευθύνσεις μνήμης όλων των προηγούμενων εντολών αποθήκευσης οι οποίες δεν

έχουν ακόμα επικυρωθεί, καθώς η εντολή φόρτωσης μπορεί να έχει εξάρτηση μνήμης από κάποια εντολή αποθήκευσης. Όπως αναφέρθηκε προηγούμενα στο κεφάλαιο αυτό, η εικασία διευθύνσεων μνήμης μπορεί να χρησιμοποιηθεί για την αντιμετώπιση του περιορισμού αυτού.

3.3 Περιορισμοί του ILP σε Ρεαλιστικούς Επεξεργαστές

Στην ενότητα αυτή, εξετάζουμε την απόδοση των επεξεργαστών που διαθέτουν υψηλό επίπεδο υποστήριξης υλικού το οποίο είναι ισοδύναμο ή καλύτερο του διαθέσιμου υλικού κατά το 2006 ή του υλικού που πρόκειται να διατεθεί τα επόμενα χρόνια. Συγκεκριμένα, θεωρούμε ότι ισχύουν οι παρακάτω σταθερές προϋποθέσεις:

1. Έκδοση έως και 64 εντολών ανά κύκλο ρολογιού χωρίς να υφίστανται περιορισμοί έκδοσης ή έκδοση σχεδόν δεκαπλάσιου αριθμού εντολών από τον αριθμό των εντολών που μπορούσε να εκδίδει ο επεξεργαστής με το μεγαλύτερο εύρος έκδοσης κατά το 2005. Όπως αναφέραμε και προηγουμένως, οι πρακτικές δυσκολίες που επιφέρει η έκδοση μεγάλου εύρους στο ρυθμό του ρολογιού, στην πολυπλοκότητα της λογικής και στην ισχύ ενδέχεται να αποτελούν τον πλέον σημαντικό περιοριστικό παράγοντα αξιοποίησης του ILP.
2. Υπαρξη ενός επιλεκτικού μηχανισμού πρόβλεψης με 1Κ καταχωρίσεις και ενός μηχανισμού πρόβλεψης επιστροφών με 16 καταχωρίσεις. Αυτός ο μηχανισμός πρόβλεψης μπορεί να συγκριθεί με τους καλύτερους μηχανισμούς πρόβλεψης κατά το 2005. Ο μηχανισμός πρόβλεψης δεν αποτελεί βασικό ανασχετικό παράγοντα.
3. Εφαρμογή απόλυτα ακριβούς αποσαφήνισης των αναφορών στη μνήμη η οποία πραγματοποιείται δυναμικά - ο στόχος αυτός είναι φιλόδοξος, ωστόσο είναι υλοποιήσιμος ίσως είτε σε παράθυρα με μικρό μέγεθος (και ως εκ τούτου σε μικρούς ρυθμούς έκδοσης και μικρούς απομονωτές φόρτωσης - αποθήκευσης) είτε διαμέσου ενός μηχανισμού πρόβλεψης εξαρτήσεων μνήμης.
4. Μετονομασία καταχωρητών και χρήση 64 επιπρόσθετων καταχωρητών ακεραίων και 64 επιπρόσθετων καταχωρητών FP, μεγέθη τα οποία μπορούν να συγκριθούν με αυτά του IBM Power 5.

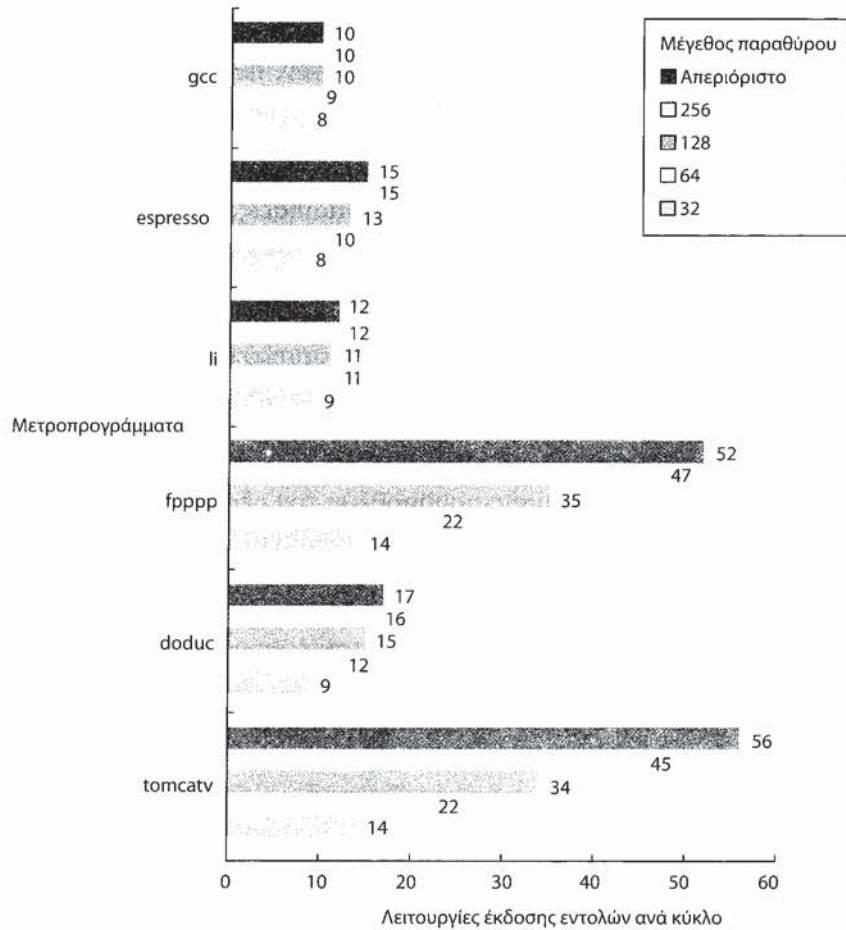
Το Σχήμα 3.7 παρουσιάζει τα αποτελέσματα που προέκυψαν από τη χρήση τη διάταξης αυτής κατά τη μεταβολή του μεγέθους του παραθύρου. Η διάταξη

αυτή είναι πιο πολύπλοκη και δαπανηρή από οποιαδήποτε υφιστάμενη υλοποίηση, ιδίως σε ό,τι αφορά τον αριθμό των εντολών που εκδίδονται, ο οποίος είναι περισσότερες από 10 φορές υψηλότερος από το μεγαλύτερο διαθέσιμο αριθμό εντολών που μπορούσαν να εκδοθούν σε οποιονδήποτε επεξεργαστή κατά το 2005. Παρ' όλα αυτά, μας παρέχει πληροφορίες σχετικά με τα αποτελέσματα που μπορούν να αποφέρουν οι μελλοντικές υλοποιήσεις. Τα δεδομένα στα σχήματα αυτά πιθανότατα μπορούν να θεωρηθούν ιδιαίτερα αισιόδοξα για ακόμα ένα λόγο. Δεν υπάρχει κανένας περιορισμός έκδοσης μεταξύ των 64 εντολών: Μπορούν όλες να αποτελούν αναφορές στη μνήμη. Κανείς στο κοντινό μέλλον δεν πρόκειται καν να διανοηθεί να προσθέσει σε κάποιον επεξεργαστή τη δυνατότητα αυτή. Δυστυχώς, είναι αρκετά δύσκολος ο περιορισμός της απόδοσης ενός επεξεργαστή χρησιμοποιώντας λογικούς περιορισμούς έκδοσης. Και αυτό διότι όχι μόνο ο χώρος των πιθανοτήτων είναι ιδιαίτερα μεγάλος, αλλά, συγχρόνως, η ύπαρξη των περιορισμών έκδοσης προϋποθέτει την αξιολόγηση του παραλληλισμού με τη βοήθεια ενός ακριβούς χρονοδρομολογητή εντολών, γεγονός που καθιστά ιδιαίτερα υψηλό το κόστος των επεξεργασιών με μεγάλο αριθμό εντολών που εκδίδονται.

Επιπλέον, ας έχουμε κατά νου ότι, κατά την ερμηνεία των αποτελεσμάτων αυτών, δεν έχουν ληφθεί υπόψη οι αστοχίες της κρυφής μνήμης και οι λανθάνοντες χρόνοι που δεν σχετίζονται με τις μονάδες, στοιχεία τα οποία στην πραγματικότητα έχουν σημαντική επίδραση.

Η πιο εντυπωσιακή παρατήρηση του Σχήματος 3.7 αφορά το γεγονός ότι στην περίπτωση που εφαρμόζονται οι ρεαλιστικοί περιορισμοί του επεξεργαστή, οι οποίοι παρατέθηκαν παραπάνω, η επίδραση που έχει το μέγεθος του παραθύρου στα προγράμματα ακεραίων δεν είναι τόσο ισχυρή όσο στην περίπτωση των προγραμμάτων FP. Το αποτέλεσμα αυτό καταδεικνύει τη βασική διαφορά των δύο αυτών κατηγοριών προγραμμάτων. Η διαθεσιμότητα του παραλληλισμού επιπέδου βρόχου σε δύο από τα προγράμματα FP συνεπάγεται ότι ο βαθμός αξιοποίησης του ΙΛΡ είναι υψηλότερος, ωστόσο για τα προγράμματα ακεραίων άλλοι είναι οι παράγοντες - όπως η πρόβλεψη διακλάδωσης, η μετονομασία καταχωρητών και ο χαμηλότερος βαθμός αρχικού παραλληλισμού - που αποτελούν σημαντικούς περιορισμούς. Η παρατήρηση αυτή έχει ζωτική σημασία εξαιτίας της αυξημένης έμφασης που έχει δοθεί στην απόδοση των ακεραίων τα τελευταία έτη. Πράγματι, το μεγαλύτερο μέρος της ανάπτυξης της αγοράς την τελευταία δεκαετία - στην επεξεργασία συναλλαγών, στους εξυπηρετητές του διαδικτύου κ.ο.κ. - είχε άμεση εξάρτηση από την απόδοση των ακεραίων και όχι από την απόδοση της κινητής υποδιαστολής. Όπως θα διαπιστώσουμε στην επόμενη ενότητα, τα επίπεδα της πραγματικής απόδοσης των ρεαλιστικών επεξεργασιών του 2005 είναι αρκετά χαμηλότερα από αυτά που παρουσιάζονται στο Σχήμα 3.7.

Με δεδομένη τη δυσκολία αύξησης του ρυθμού των εντολών χρησιμοποιώντας ρεαλιστικές σχεδιάσεις υλικού, οι σχεδιαστές αντιμετωπίζουν μια πρόκληση



Σχήμα 3.7 Παρουσίαση του εύρους του διαθέσιμου παραλληλισμού σε σύγκριση με το μέγεθος παραθύρου για πληθώρα προγραμμάτων ακεραίων και κινητής υποδιαστολής, με δυνατότητα έκδοσης 64 αυθαίρετων ενεργειών έκδοσης εντολών ανά κύκλο. Παρόλο που ο αριθμός των καταχωρητών μετονομασίας είναι μικρότερος από το μέγεθος του παραθύρου, το γεγονός ότι όλες οι λειτουργίες διαθέτουν μηδενικό λανθάνοντα χρόνο, καθώς και το ότι ο αριθμός των καταχωρητών μετονομασίας είναι ίσος με το εύρος της έκδοσης επιτρέπουν στον επεξεργαστή να αξιοποιεί τον παραλληλισμό εντός ολόκληρου του παραθύρου. Στις πραγματικές υλοποιήσεις, το μέγεθος του παραθύρου και ο αριθμός των καταχωρητών μετονομασίας πρέπει να είναι ισορροπημένοι, έτσι ώστε να παρεμποδίζεται ο περιορισμός του ρυθμού έκδοσης από κάποιον από αυτούς τους παράγοντες.

όσον αφορά τη λήψη της απόφασης σχετικά με τον τρόπο της καλύτερης δυνατής χρήσης των περιορισμένων διαθέσιμων πόρων των ολοκληρωμένων κυκλωμάτων. Μία από τις πιο ενδιαφέρουσες αλληλοεξαρτώμενες σχέσεις είναι αυτή ανάμεσα στην προσέγγιση χρήσης πιο απλών επεξεργαστών με μεγάλες κρυφές μνήμες

και υψηλούς ρυθμούς ρολογιού και στην προσέγγιση επικέντρωσης της προσοχής στον παραλληλισμό επιπέδου εντολών χρησιμοποιώντας χαμηλότερους ρυθμούς ρολογιού και μικρότερες κρυφές μνήμες. Το παρακάτω παράδειγμα απεικονίζει τις προκλήσεις αυτές.

Παράδειγμα: Ας εξετάσουμε τους παρακάτω τρεις υποθετικούς, αλλά όχι ασυνήθιστους, επεξεργαστές, στους οποίους έχουμε εκτελέσει το μετροπρόγραμμα gcc του SPEC:

1. Μια απλή στατική σωλήνωση διπλής έκδοσης της MIPS που εκτελείται με ρυθμό ρολογιού της τάξης των 4 GHz και διαθέτει CPI διασωλήνωσης ίσο με 0.8. Ο επεξεργαστής αυτός διαθέτει σύστημα κρυφής μνήμης με 0.005 αστοχίες ανά εντολή.
2. Έναν επεξεργαστή που διαθέτει διασωλήνωση μεγάλου βάθους διπλής έκδοσης της MIPS με ελαφρά μικρότερες κρυφές μνήμες και ρυθμό ρολογιού της τάξης των 5 GHz. Το CPI της διασωλήνωσης του επεξεργαστή ισούται με 1.0, ενώ οι μικρότερες κρυφές μνήμες σημειώνουν κατά μέσο όρο 0,0055 αστοχίες ανά εντολή.
3. Έναν υποθετικό υπερβαθμωτό επεξεργαστή με παράθυρο 64 καταχωρίσεων. Ο επεξεργαστής αυτός είναι σε θέση να σημειώνει κατά το ήμισυ τον ιδανικό ρυθμό έκδοσης για αυτό το μέγεθος παραθύρου (χρησιμοποιήστε τα δεδομένα του Σχήματος 3.7). Ο επεξεργαστής αυτός διαθέτει τις μικρότερες κρυφές μνήμες, κάτι που έχει ως αποτέλεσμα να σημειώνονται 0.01 αστοχίες ανά εντολή, ωστόσο ο επεξεργαστής αποκρύβει μέσω της δυναμικής χρονοδρομολόγησης το 25% της ποινής κάθε αστοχίας. Ο επεξεργαστής αυτός έχει ρυθμό ρολογιού της τάξης των 2.5 GHz.

Υποθέτουμε ότι ο χρόνος της κύριας μνήμης (ο οποίος καθορίζει την ποινή αστοχίας) ανέρχεται σε 50 ns. Ας προσδιορίσουμε τη σχετική απόδοση των τριών αυτών επεξεργαστών.

Απάντηση: Πρώτον, χρησιμοποιούμε την ποινή αστοχίας και τις πληροφορίες του ρυθμού αστοχίας για τον υπολογισμό του βαθμού στον οποίο επηρεάζεται το CPI από τις αστοχίες της κρυφής μνήμης κάθε διάταξης. Αυτό το πραγματοποιούμε με τη βοήθεια του παρακάτω τύπου:

$$\text{CPI κρυφής μνήμης} = \text{Αστοχίες ανά εντολή} \times \text{Ποινή αστοχίας}$$

Χρειάζεται να υπολογίσουμε τις ποινές αστοχίας κάθε συστήματος:

$$\text{Ποινή αστοχίας} = \frac{\text{Χρόνος προσπέλασης μνήμης}}{\text{Κύκλος ρολογιού}}$$

Οι χρόνοι του κύκλου ρολογιού για τους επεξεργαστές ισούνται με 250 ps, 200 ps και 400 ps αντίστοιχα. Επομένως, οι ποινές αστοχίας ισούνται με

$$\begin{aligned} \text{Ποινή αστοχίας}_1 &= \frac{50\text{ns}}{250\text{ps}} = 200 \text{ κύκλοι} \\ \text{Ποινή αστοχίας}_2 &= \frac{50\text{ns}}{200\text{ps}} = 250 \text{ κύκλοι} \\ \text{Ποινή αστοχίας}_3 &= \frac{0.75 \times 50\text{ns}}{400\text{ps}} = 94 \text{ κύκλοι} \end{aligned}$$

Η εφαρμογή των παραπάνω αποτελεσμάτων σε κάθε κρυφή μνήμη έχει ως ακολούθως:

$$\begin{aligned} \text{CPI κρυφής μνήμης}_1 &= 0.005 \times 200 = 1.0 \\ \text{CPI κρυφής μνήμης}_2 &= 0.005 \times 250 = 1.4 \\ \text{CPI κρυφής μνήμης}_3 &= 0.005 \times 94 = 0.94 \end{aligned}$$

Γνωρίζουμε το βαθμό συνεισφοράς της διασωλήνωσης στο CPI για όλες τις περιπτώσεις εκτός από αυτήν του επεξεργαστή 3. Το CPI της διασωλήνωσης του δίνεται από τον παρακάτω τύπο:

$$\text{CPI διασωλήνωσης}_3 = \frac{1}{\text{Ρυθμός έκδοσης}} = \frac{1}{9 \times 0.5} = \frac{1}{4.5} = 0.22$$

Πλέον μπορούμε να υπολογίσουμε το CPI κάθε επεξεργαστή προσθέτοντας το βαθμό συνεισφοράς της διασωλήνωσης και της κρυφής μνήμης στο CPI.

$$\begin{aligned} \text{CPI}_1 &= 0.8 + 1.0 = 1.8 \\ \text{CPI}_2 &= 1.0 + 1.4 = 2.4 \\ \text{CPI}_3 &= 0.22 + 0.94 = 1.16 \end{aligned}$$

Καθώς σε όλες τις περιπτώσεις έχουμε την ίδια αρχιτεκτονική, μπορούμε να συγκρίνουμε τους ρυθμούς εκτέλεσης εντολών σε εκατομμύρια εντολές ανά δευτερόλεπτο (million instructions per second, MIPS) για τον προσδιορισμό της σχετικής απόδοσης:

$$\begin{aligned} \text{Ρυθμός εκτέλεσης εντολών} &= \frac{\text{CR}}{\text{CPI}} \\ \text{Ρυθμός εκτέλεσης εντολών}_1 &= \frac{4000 \text{ MHz}}{1.8} = 2222 \text{ MIPS} \\ \text{Ρυθμός εκτέλεσης εντολών}_2 &= \frac{5000 \text{ MHz}}{2.4} = 2083 \text{ MIPS} \\ \text{Ρυθμός εκτέλεσης εντολών}_3 &= \frac{2.500 \text{ MHz}}{1.16} = 2155 \text{ MIPS} \end{aligned}$$

Στο παράδειγμα αυτό, ο απλός στατικός υπερβαθμωτός επεξεργαστής διπλής έκδοσης φαίνεται πως αποτελεί την καλύτερη λύση. Στην πράξη, η απόδοση εξαρτάται τόσο από το CPI όσο και από το ρυθμό ρολογιού.

Πέρα από τα Όρια της Μελέτης Αυτής

Όπως κάθε μελέτη περιορισμών, έτσι και η μελέτη που εξετάσαμε στην ενότητα αυτή διαθέτει τους δικούς της περιορισμούς. Διακρίνουμε τους περιορισμούς αυτούς σε δύο κατηγορίες: στους περιορισμούς που προκύπτουν ακόμα και στην περίπτωση του απόλυτα ακριβούς υποθετικού επεξεργαστή και στους περιορισμούς που προκύπτουν για ένα ή περισσότερα ρεαλιστικά μοντέλα. Βεβαίως, όλοι οι περιορισμοί της πρώτης κατηγορίας ισχύουν και για τη δεύτερη. Οι πιο σημαντικοί περιορισμοί που υφίστανται ακόμα και στην περίπτωση του απόλυτα ακριβούς μοντέλου είναι:

1. *Κίνδυνοι WAR και WAW μέσω της μνήμης:* Η μελέτη εξάλειψε την εμφάνιση των κινδύνων WAR και WAW μέσω της μετονομασίας καταχωρητών, ωστόσο οι κίνδυνοι αυτοί εξακολουθούν να υφίστανται κατά τη χρήση της μνήμης. Παρόλο που εκ πρώτης άποψης μπορεί να θεωρείται ότι αυτές οι περιπτώσεις είναι σπάνιες (ιδίως σε ό,τι αφορά τους κινδύνους WAW), οι κίνδυνοι προκύπτουν εξαιτίας της ανάθεσης των πλαισίων στοίβας. Οι διαδικασίες που έχουν κληθεί χρησιμοποιούν τις θέσεις μνήμης κάποιας προηγούμενης διαδικασίας που βρισκόταν στη στοίβα και αυτό μπορεί να οδηγήσει σε κινδύνους WAR και WAW οι οποίοι είναι περισσότερο περιοριστικοί απ' ό,τι χρειάζεται. Οι Austin και Sohi [1992] εξετάζουν το ζήτημα αυτό.
2. *Περιττές εξαρτήσεις:* Όταν υφίσταται απεριόριστος αριθμός καταχωρητών, αφαιρούνται όλες οι εξαρτήσεις δεδομένων εκτός από τις πραγματικές εξαρτήσεις δεδομένων καταχωρητών. Ωστόσο, υπάρχουν εξαρτήσεις που προκύπτουν είτε από την επανεμφάνιση εντολών είτε από κανόνες που

αφορούν την παραγωγή κώδικα, οι οποίοι έχουν ως αποτέλεσμα την εισαγωγή περιττών πραγματικών εξαρτήσεων δεδομένων. Ένα από αυτά τα παραδείγματα είναι εξάρτηση της μεταβλητής ελέγχου σε έναν απλό βρόχο do: Καθώς η μεταβλητή ελέγχου αυξάνεται σε κάθε επανάληψη του βρόχου, ο βρόχος περιλαμβάνει τουλάχιστον μία εξάρτηση. Όπως μπορεί να διαπιστώσει κανείς ανατρέχοντας στο Παράρτημα Ζ, το ξεδίπλωμα του βρόχου και η εντατική αλγεβρική βελτιστοποίηση μπορούν να οδηγήσουν στην αφαίρεση αυτού του είδους των εξαρτώμενων υπολογισμών. Η μελέτη του Wall περιλαμβάνει περιορισμένο αριθμό τέτοιων βελτιστοποιήσεων, ωστόσο η εντατική εφαρμογή τους μπορεί να οδηγήσει σε αύξηση του εύρους του ILP. Επιπλέον, συγκεκριμένοι κανόνες που αφορούν την παραγωγή του κώδικα εισάγουν περιττές εξαρτήσεις, ιδίως με τη χρήση των καταχωρητών διεύθυνσης επιστροφής και του καταχωρητή δείκτη στοιβάς (του οποίου η τιμή αυξάνεται και μειώνεται κατά την ακολουθία κλήσης/επιστροφής). Ο Wall έχει εξαλείψει τις επιπτώσεις του καταχωρητή διεύθυνσης επιστροφής, ωστόσο η χρήση του δείκτη στοιβάς μπορεί να προκαλέσει «περιττές» εξαρτήσεις. Οι Postiff et al. [1999] διερεύνησαν τα πλεονεκτήματα που προέκυπταν από την εξάλειψη του περιορισμού αυτού.

3. *Αντιμετώπιση του περιορισμού της ροής δεδομένων*: Αν η πρόβλεψη τιμής λειτουργούσε διαθέτοντας απόλυτη ακρίβεια, τότε θα ήταν σε θέση να υπερκεράσει τον περιορισμό της ροής δεδομένων. Μέχρι σήμερα, κανένα από τα 50 άρθρα που έχουν καταπιαστεί με το θέμα αυτό δεν έχει καταφέρει να επιφέρει κάποια σημαντική βελτίωση του ILP, όταν χρησιμοποιείται κάποια ρεαλιστική προσέγγιση πρόβλεψης. Όπως είναι φανερό, η απόλυτα ακριβής πρόβλεψη τιμής δεδομένων θα οδηγούσε ουσιαστικά σε απεριόριστο παραλληλισμό, καθώς όλες οι τιμές κάθε εντολής θα μπορούσαν να προβλεφθούν εκ των προτέρων.

Αρκετές ιδέες έχουν διατυπωθεί όσον αφορά την αποκάλυψη του ILP σε μεγαλύτερο βαθμό σε διατάξεις επεξεργαστών που είναι υποδεέστερες του απόλυτα ακριβούς επεξεργαστή. Σχετικό παράδειγμα αποτελεί η εφαρμογή της εικασίας σε πολλαπλές διαδρομές. Η ιδέα αυτή εξετάστηκε από τους Lam και Wilson [1992] και διερευνήθηκε στο πλαίσιο της μελέτης της ενότητας αυτής. Μέσω της εφαρμογής της εικασίας σε πολλαπλές διαδρομές μειώνεται το κόστος της ανάκαμψης από εσφαλμένες προβλέψεις, ενώ συγχρόνως αποκαλύπτεται υψηλότερο εύρος παραλληλισμού. Η αξιολόγηση της προσέγγισης αυτής μπορεί να πραγματοποιηθεί μόνο για την εξέταση περιορισμένου αριθμού διακλαδώσεων, διότι οι πόροι υλικού που απαιτούνται αυξάνονται με γεωμετρική πρόοδο. Ο Wall [1993] παρέχει δεδομένα για την εφαρμογή της εικασίας και προς τις δύο κατευθύνσεις μέχρι και για οκτώ διακλαδώσεις. Όλες οι εμπορικές σχεδιάσεις, έχοντας γνώση του κόστους της εξέτασης και των δύο κατευθύνσεων και του

γεγονότος ότι μία από αυτές τις κατευθύνσεις θα απορριφθεί (καθώς και του αυξανόμενου αριθμού περιπτώσεων υπολογισμών όσο αυτού του είδους η διαδικασία ακολουθείται σε πολλαπλές διακλαδώσεις) έχουν αντιθέτως αφιερώσει τη χρήση επιπρόσθετων μονάδων υλικού για την καλύτερη εφαρμογή της εικασίας στην ορθή διαδρομή.

Είναι πολύ σημαντικό να κατανοήσει κανείς ότι κανένας από τους περιορισμούς της ενότητας αυτής δεν είναι θεμελιώδης, υπό την έννοια ότι η αντιμετώπισή τους δεν απαιτεί την αλλαγή των νόμων της φυσικής! Αντιθέτως, αποτελούν πρακτικούς περιορισμούς οι οποίοι υποδηλώνουν την ύπαρξη μερικών απαγορευτικών εμποδίων που αποτελούν τροχοπέδη για την περαιτέρω αξιοποίηση του ILP. Οι περιορισμοί αυτοί - είτε πρόκειται για το μέγεθος του παραθύρου και την ανίχνευση της ψευδωνυμίας είτε για την πρόβλεψη διακλάδωσης - αντικατοπτρίζουν προκλήσεις τις οποίες οι σχεδιαστές και οι ερευνητές καλούνται να αντιμετωπίσουν! Όπως θα διαπιστώσουμε στην Ενότητα 3.6, οι επιπλοκές που επιφέρουν οι περιορισμοί του ILP σε συνδυασμό με το κόστος της υλοποίησης προσεγγίσεων έκδοσης εντολών μεγαλύτερου εύρους φαίνεται πως έχουν δημιουργήσει μερικούς σημαντικούς περιορισμούς όσον αφορά την αξιοποίηση του ILP.

3.4 Πολυεπίπεδα Ζητήματα: Σύγκριση Εικασίας Υλικού με Εικασία Λογισμικού

Οι ενότητες με τίτλο «Πολυεπίπεδα Ζητήματα» εξετάζουν ζητήματα που σχετίζονται με το αντικείμενο διαφορετικών κεφαλαίων. Τα επόμενα κεφάλαια περιλαμβάνουν τέτοιες ενότητες.

Οι προσεγγίσεις εντατικής χρήσης υλικού για την εφαρμογή της εικασίας που αναλύθηκαν στο προηγούμενο κεφάλαιο και οι προσεγγίσεις λογισμικού που περιγράφονται στο Παράρτημα Ζ παρέχουν εναλλακτικές προσεγγίσεις για την αξιοποίηση του ILP. Παρακάτω παρατίθενται μερικές από τις αλληλοεξαρτώμενες σχέσεις και τους περιορισμούς που σχετίζονται με τις προσεγγίσεις αυτές:

- Προκειμένου να είναι δυνατή η εκτεταμένη εφαρμογή της εικασίας, πρέπει να είναι εφικτή η αποσαφήνιση των αναφορών στη μνήμη. Η δυνατότητα αυτή είναι δύσκολο να παρασχεθεί κατά το χρόνο μεταγλώττισης για προγράμματα ακεραίων που περιέχουν δείκτες. Στις προσεγγίσεις υλικού, η δυναμική αποσαφήνιση των διευθύνσεων της μνήμης κατά το χρόνο εκτέλεσης πραγματοποιείται χρησιμοποιώντας τεχνικές που εξετάσαμε προηγούμενα για τον αλγόριθμο του Tomasulo. Η αποσαφήνιση αυτή μας επιτρέπει κατά το χρόνο εκτέλεσης να μεταφέρουμε τις εντολές φόρτωσης μετά από τις εντολές αποθήκευσης. Η υποστήριξη όσον αφορά τις υποθετικές αναφορές μνήμης μπορεί να βοηθήσει στην αντιμετώπιση

του συντηρητισμού του μεταγλωττιστή, ωστόσο αν αυτού του είδους οι προσεγγίσεις δεν χρησιμοποιηθούν προσεκτικά, η επιβάρυνση που προκύπτει εξαιτίας των μηχανισμών ανάκαμψης μπορεί να εκμηδενίσει τα οφέλη που προκύπτουν.

- Η εικασία υλικού λειτουργεί καλύτερα όταν η ροή δεδομένων δεν μπορεί να προβλεφθεί, καθώς και όταν η πρόβλεψη διακλάδωσης μέσω υλικού είναι ανώτερη από την πρόβλεψη διακλάδωσης μέσω λογισμικού, που πραγματοποιείται κατά το χρόνο μεταγλώττισης. Οι ιδιότητες αυτές ισχύουν για πολλά προγράμματα ακεραίων. Για παράδειγμα, οι στατικοί μηχανισμοί πρόβλεψης που λειτουργούν σε ικανοποιητικό βαθμό διαθέτουν ρυθμό εσφαλμένων προβλέψεων της τάξης του 16% για τέσσερα από τα πιο σημαντικά προγράμματα ακεραίων του SPEC92, ενώ οι μηχανισμοί πρόβλεψης υλικού διαθέτουν ρυθμό εσφαλμένων προβλέψεων που είναι μικρότερος του 10%. Η διαφορά αυτή είναι σημαντική, επειδή οι υποθετικές εντολές ενδέχεται να επιβραδύνουν τη διαδικασία υπολογισμού σε περίπτωση που η πρόβλεψη είναι λανθασμένη. Αποτέλεσμα της παραπάνω διαφοράς είναι το γεγονός ότι ακόμα και οι στατικά χρονοδρομολογημένοι επεξεργαστές συνήθως περιλαμβάνουν δυναμικούς μηχανισμούς πρόβλεψης διακλάδωσης.
- Η εικασία υλικού διαθέτει απόλυτα ακριβές μοντέλο εξαιρέσεων, ακόμα και για τις υποθετικές εντολές. Οι πρόσφατες προσεγγίσεις λογισμικού έχουν προσθέσει ειδική υποστήριξη, έτσι ώστε να παρέχεται και αυτή η δυνατότητα.
- Η εικασία υλικού δεν απαιτεί κώδικα επανόρθωσης ή καταγραφής, ο οποίος απαιτείται στην περίπτωση που χρησιμοποιούνται φιλόδοξοι μηχανισμοί εικασίας λογισμικού.
- Οι προσεγγίσεις που βασίζονται σε μεταγλωττιστή μπορεί να επωφεληθούν από το γεγονός ότι έχουν τη δυνατότητα να έχουν εικόνα μεγάλου τμήματος της ακολουθίας του κώδικα, κάτι που έχει ως αποτελέσματα την καλύτερη χρονοδρομολόγηση του κώδικα από την περίπτωση που ακολουθείται προσέγγιση που βασίζεται εξ ολοκλήρου στο υλικό.
- Η εικασία υλικού σε συνδυασμό με τη δυναμική χρονοδρομολόγηση δεν απαιτεί τη χρήση διαφορετικών ακολουθιών κώδικα για την επίτευξη υψηλής απόδοσης σε διαφορετικές υλοποιήσεις της αρχιτεκτονικής. Το όφελος αυτό, παρόλο που είναι πολύ δύσκολο να ποσοτικοποιηθεί, ενδέχεται να αποτελεί την πιο σημαντική παράμετρο σε βάθος χρόνου. Ενδιαφέρον παρουσιάζει το γεγονός ότι αυτό αποτέλεσε ένα από τα βασικά κίνητρα του IBM 390/91. Από την άλλη πλευρά, οι πιο πρόσφατες ρητά παράλληλες

αρχιτεκτονικές, όπως η IA-64, έχουν προσαρμοστεί ώστε να μειώνεται η εγγενής εξάρτηση υλικού που υφίσταται στις ακολουθίες κώδικα.

Το βασικό μειονέκτημα της υποστήριξης της εικασίας μέσω υλικού είναι η πολυπλοκότητα και οι επιπρόσθετοι πόροι υλικού που απαιτούνται. Αυτό το κόστος του υλικού πρέπει να αξιολογηθεί σε σύγκριση τόσο με την πολυπλοκότητα που θα έπρεπε να διαθέτει κάποιος μεταγλωττιστής σε προσέγγιση λογισμικού όσο και με και τη χρησιμότητα και το βαθμό απλοποίησης των επεξεργασιών που βασίζονται σε τέτοιου είδους μεταγλωττιστή. Θα επιστρέψουμε στο ζήτημα αυτό στις συμπερασματικές παρατηρήσεις.

Μερικοί σχεδιαστές έχουν επιχειρήσει να συνδυάσουν τη δυναμική προσέγγιση με τη προσέγγιση χρήσης μεταγλωττιστή, έτσι ώστε να αποκομίσουν τα μέγιστα δυνατά οφέλη και από τις δύο περιπτώσεις. Αυτού του είδους ο συνδυασμός μπορεί να παραγάγει σημαντικές δυσδιάκριτες αλληλεπιδράσεις. Για παράδειγμα, αν η μεταφορά υπό συνθήκη (conditional move) συνδυάζεται με τη μετονομασία καταχωρητών, τότε προκύπτει ένα δυσδιάκριτο παράπλευρο φαινόμενο. Οι μεταφορές υπό συνθήκη που έχουν ακυρωθεί εξακολουθούν να χρειάζεται να αντιγράψουν κάποια τιμή στον καταχωρητή προορισμού, καθώς αυτός είχε μετονομαστεί νωρίτερα εντός της διασωλήνωσης των εντολών. Αυτές οι δυσδιάκριτες αλληλεπιδράσεις περιπλέκουν τη διαδικασία σχεδίασης και επαλήθευσης, ενώ συγχρόνως ενδέχεται να προκαλούν μείωση της απόδοσης.

3.5 Πολυνημάτωση: Χρήση της Υποστήριξης του ILP για την Αξιοποίηση του Παραλληλισμού Επιπέδου Νημάτων

Παρόλο που η αύξηση της απόδοσης μέσω της χρήσης του ILP έχει το σημαντικό πλεονέκτημα ότι δεν είναι σε μεγάλο βαθμό αντιληπτή από τον προγραμματιστή, όπως έχουμε διαπιστώσει, ο ILP μπορεί να περιορίζεται σημαντικά ή να είναι δύσκολο να αξιοποιηθεί από μερικές εφαρμογές. Επιπλέον, μπορεί να υπάρχει μεγάλος βαθμός παραλληλισμού που επιτυγχάνεται με φυσικό τρόπο σε υψηλότερα επίπεδα της εφαρμογής, ο οποίος δεν μπορεί να αξιοποιηθεί μέσω των προσεγγίσεων που εξετάστηκαν στο κεφάλαιο αυτό. Για παράδειγμα, τα online συστήματα επεξεργασίας συναλλαγών διαθέτουν από τη φύση τους παραλληλισμό μεταξύ των πολλαπλών ερωτημάτων (queries) και ενημερώσεων (updates) που πραγματοποιούνται μέσω των αιτήσεων. Η επεξεργασία αυτών των ερωτημάτων και των ενημερώσεων μπορεί να πραγματοποιηθεί ως επί το πλείστον παράλληλα, καθώς τα ερωτήματα και οι ενημερώσεις είναι σε μεγάλο βαθμό ανεξάρτητα μεταξύ τους. Βεβαίως, πολλές επιστημονικές εφαρμογές διαθέτουν από τη φύση τους παραλληλισμό, καθώς μοντελοποιούν την τριδιάστατη παράλληλη δομή της φύσης, δομή η οποία μπορεί να αξιοποιηθεί στο πλαίσιο προσομοιώσεων.

Αυτό το υψηλότερο επίπεδο παραλληλισμού ονομάζεται *παραβήθισμός επιπέδου νημάτων* (thread-level parallelism, TLP), διότι είναι σε λογικό επίπεδο δομημένος ως διαφορετικά νήματα εκτέλεσης. Το *νήμα* (thread) αποτελεί μια ξεχωριστή διεργασία (process) με τις δικές του εντολές και δεδομένα. Το νήμα μπορεί είτε να αναπαριστά κάποια διεργασία η οποία είναι μέρος ενός παράλληλου προγράμματος το οποίο αποτελείται από πολλαπλές διεργασίες είτε το ίδιο να αποτελεί κάποιο ανεξάρτητο πρόγραμμα. Κάθε νήμα διαθέτει όλα τα απαραίτητα στοιχεία κατάστασης (εντολές, δεδομένα, PC, κατάσταση καταχωρητών κ.ο.κ.) που επιτρέπουν την εκτέλεσή του. Σε αντίθεση με τον παραλληλισμό επιπέδου εντολών, ο οποίος αξιοποιεί έμμεσα παράλληλες λειτουργίες εντός κάποιου βρόχου ή κάποιου ευθύγραμμου τμήματος κώδικα, ο παραλληλισμός επιπέδου νημάτων αναπαρίσταται με σαφή τρόπο μέσω της χρήσης πολλαπλών νημάτων εκτέλεσης, τα οποία από τη φύση τους είναι παράλληλα.

Ο παραλληλισμός επιπέδου νημάτων αποτελεί σημαντική εναλλακτική λύση του παραλληλισμού επιπέδου εντολών, κυρίως, διότι η αξιοποίησή του απαιτεί μικρότερο κόστος από τον παραλληλισμό επιπέδου εντολών. Υπάρχουν πολλές σημαντικές εφαρμογές στις οποίες ο παραλληλισμός επιπέδου νημάτων προκύπτει με φυσικό τρόπο, κάτι που συμβαίνει σε πολλές εφαρμογές εξυπηρετητή (server applications). Σε άλλες περιπτώσεις, το λογισμικό γράφεται από μηδενική βάση, ενώ είναι εύκολο να εκφραστεί ο εγγενής παραλληλισμός, κάτι το οποίο υλοποιείται σε μερικές ενσωματωμένες εφαρμογές (embedded applications). Παρ' όλα αυτά, οι εδραιωμένες εφαρμογές μεγάλου μεγέθους, που έχουν γραφεί χωρίς να έχει ληφθεί μέριμνα για τον παραλληλισμό, αποτελούν σημαντική πρόκληση, ενώ συγχρόνως είναι ιδιαίτερα δαπανηρή η διαδικασία της εκ νέου συγγραφής τους για την αξιοποίηση του παραλληλισμού επιπέδου βρόχου. Το Κεφάλαιο 4 διερευνά τους πολυεπεξεργαστές και την υποστήριξη που παρέχουν στον παραλληλισμό επιπέδου νημάτων.

Ο παραλληλισμός επιπέδου νημάτων και ο παραλληλισμός επιπέδου εντολών αξιοποιούν δύο διαφορετικά είδη παράλληλης δομής των προγραμμάτων. Το φυσικό ερώτημα που προκύπτει αφορά αν κάποιος επεξεργαστής που είναι προσανατολισμένος να εφαρμόζει τον παραλληλισμό επιπέδου εντολών έχει τη δυνατότητα να αξιοποιεί τον παραλληλισμό επιπέδου νημάτων. Κίνητρο για το ερώτημα αυτό έχει αποτελέσει η παρατήρηση ότι σε κάποια διαδρομή δεδομένων που έχει σχεδιαστεί για την αξιοποίηση του ILP σε μεγαλύτερο βαθμό, συχνά παρατηρείται το φαινόμενο οι λειτουργικές μονάδες να είναι αδρανείς, είτε εξαιτίας διαστημάτων αδράνειας είτε εξαιτίας εξαρτήσεων που υφίστανται στον κώδικα. Μπορεί ο παραλληλισμός που υφίσταται μεταξύ των νημάτων να χρησιμοποιηθεί ως αφετηρία ανεξάρτητων εντολών οι οποίες μπορεί να διατηρούν τον επεξεργαστή απασχολημένο κατά τη διάρκεια των διαστημάτων αδράνειας; Μπορεί αυτός ο παραλληλισμός επιπέδου νημάτων να χρησιμοποιηθεί για την αξιοποίηση των λειτουργικών μονάδων οι οποίες διαφορετικά παραμένουν αδρανείς, όταν δεν υφίσταται επαρκής ILP;

Η *πολυνημάτωση* (multithreading) επιτρέπει σε πολλαπλά νήματα να διαμοιράζονται τις λειτουργικές μονάδες ενός επεξεργαστή με αλληλοεπικαλυπτόμενο τρόπο. Για να επιτραπεί ο διαμοιρασμός αυτός, ο επεξεργαστής πρέπει να αντιγράφει την ανεξάρτητη κατάσταση κάθε νήματος. Για παράδειγμα, για κάθε νήμα χρειάζεται ξεχωριστό αντίγραφο της συστοιχίας καταχωρητών, ξεχωριστός PC και ξεχωριστός πίνακας σελίδων. Ο διαμοιρασμός της ίδιας της μνήμης μπορεί να πραγματοποιείται μέσω των μηχανισμών της ιδεατής μνήμης (virtual memory), η οποία υποστηρίζει τον πολυπρογραμματισμό. Επιπλέον, το υλικό πρέπει να υποστηρίζει τη δυνατότητα αλλαγής (switch) σε κάποιο άλλο νήμα με σχετικά υψηλή ταχύτητα. Συγκεκριμένα, η αλλαγή των νημάτων πρέπει να είναι πολύ πιο αποτελεσματική από την αλλαγή διεργασιών, η οποία συνήθως απαιτεί εκατοντάδες έως χιλιάδες κύκλους του επεξεργαστή.

Υπάρχουν δύο βασικές προσεγγίσεις όσον αφορά την πολυνημάτωση. Η *λεπτόκοκκη πολυνημάτωση* (fine-grained multithreading) εναλλάσσεται μεταξύ νημάτων σε κάθε εντολή, γεγονός που προκαλεί την εναλλαγή της εκτέλεσης πολλαπλών νημάτων. Η εναλλαγή αυτή συχνά πραγματοποιείται ακολουθώντας τη μέθοδο εκ περιτροπής (round-robin), αγνοώντας οποιαδήποτε νήματα τίθενται σε αδράνεια εντός αυτού του χρονικού διαστήματος. Προκειμένου η λεπτόκοκκη πολυνημάτωση να καταστεί πρακτική, η CPU πρέπει να είναι σε θέση να αλλάζει νήματα σε κάθε κύκλο ρολογιού. Βασικό πλεονέκτημα της λεπτόκοκκης πολυνημάτωσης είναι το γεγονός ότι μπορεί να αποκρύψει τις απώλειες της ρυθμοαπόδοσης οι οποίες προκύπτουν τόσο από τα σύντομα όσο και από τα μεγάλης διάρκειας διαστήματα αδράνειας, καθώς οι εντολές των υπόλοιπων νημάτων μπορούν να εκτελούνται μόλις κάποιο νήμα τίθεται σε αδράνεια. Το κύριο μειονέκτημα της λεπτόκοκκης πολυνημάτωσης είναι το γεγονός ότι επιβραδύνει την εκτέλεση των μεμονωμένων νημάτων, καθώς κάποιο νήμα που είναι έτοιμο να εκτελεστεί χωρίς διαστήματα αδράνειας θα καθυστερήσει εξαιτίας εντολών άλλων νημάτων.

Η *χονδρόκοκκη πολυνημάτωση* (coarse-grained) εφευρέθηκε ως εναλλακτική λύση της λεπτόκοκκης πολυνημάτωσης. Η χονδρόκοκκη πολυνημάτωση προβαίνει σε αλλαγή των νημάτων μόνο όταν προκύπτουν δαπανηρά διαστήματα αδράνειας, όπως αστοχίες της κρυφής μνήμης του επιπέδου 2. Η διαφοροποίηση αυτή μας απαλλάσσει από την ανάγκη να εφαρμόσουμε ουσιαστικά ελεύθερη αλλαγή νημάτων, ενώ είναι πολύ λιγότερο πιθανό να επιβραδύνει τον επεξεργαστή, καθώς η έκδοση των εντολών των υπόλοιπων νημάτων πραγματοποιείται μόνο όταν κάποιο νήμα τίθεται σε αδράνεια υψηλού κόστους.

Παρ' όλα αυτά, η χονδρόκοκκη πολυνημάτωση έχει ένα πολύ σημαντικό μειονέκτημα: Διαθέτει περιορισμένες δυνατότητες αντιμετώπισης των απωλειών της ρυθμοαπόδοσης, ιδίως αυτών που προέρχονται από τα διαστήματα αδράνειας μικρότερης διάρκειας. Ο περιορισμός αυτός προκύπτει από το κόστος εκκίνησης της διασωλήνωσης της χονδρόκοκκης πολυνημάτωσης. Επειδή η CPU που υποστηρίζει τη χονδρόκοκκη πολυνημάτωση εκδίδει εντολές από ένα μόνο νή-

μα, η διασωλήνωση πρέπει να αδειάζει ή να εκκενώνεται, όταν προκύπτει κάποιο διάστημα αδράνειας. Το νέο νήμα που ξεκινά να εκτελείται μετά το διάστημα αδράνειας πρέπει να γεμίσει τη διασωλήνωση πριν οι εντολές να είναι σε θέση να ολοκληρωθούν. Εξαιτίας αυτής της επιβάρυνσης της εκκίνησης, η χονδρόκοκκη πολυνημάτωση είναι πολύ περισσότερο χρήσιμη για τη μείωση της ποινής των διαστημάτων αδράνειας υψηλού κόστους, περίπτωση κατά την οποία ο χρόνος που χρειάζεται για να γεμίσει εκ νέου η διασωλήνωση είναι αμελητέα ποσότητα μπροστά στο χρόνο που απαιτούν τα διαστήματα αδράνειας.

Η επόμενη υποενότητα διερευνά μια παραλλαγή της λεπτόκοκκης πολυνημάτωσης η οποία παρέχει τη δυνατότητα στους υπερβαθμισμένους επεξεργαστές να αξιοποιούν την πολυνημάτωση με ολοκληρωμένο και αποδοτικό τρόπο. Στο Κεφάλαιο 4, επιστρέφουμε στο ζήτημα της πολυνημάτωσης, όταν εξετάζουμε την εφαρμογή της με πολλαπλές CPUs εντός ενός chip.

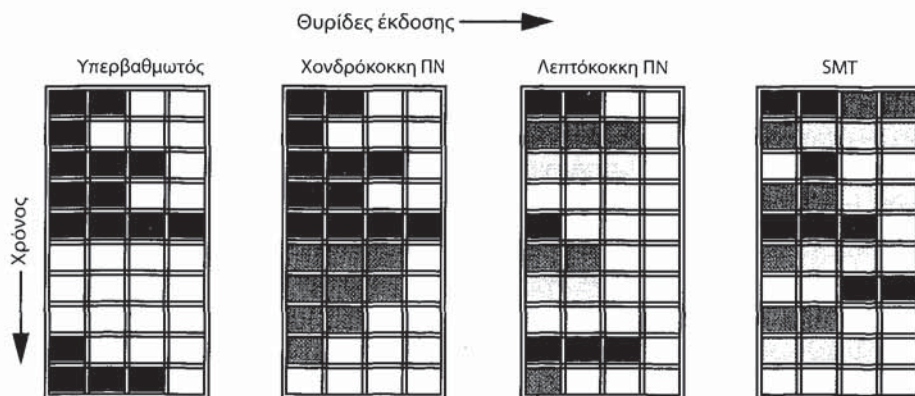
Ταυτόχρονη Πολυνημάτωση: Μετατροπή του Επιπέδου Παράλληλισμού Νημάτων σε Παράλληλισμό Επιπέδου Εντολών

Η ταυτόχρονη πολυνημάτωση (simultaneous multithreading, SMT) αποτελεί παραλλαγή της πολυνημάτωσης που χρησιμοποιεί τους πόρους των δυναμικά χρονοδρομολογημένων επεξεργαστών πολλαπλής έκδοσης για την ταυτόχρονη αξιοποίηση του TLP και του ILP. Αυτό που έχει αποτελέσει το βασικό κίνητρο για την SMT είναι το γεγονός ότι οι σύγχρονοι επεξεργαστές πολλαπλής έκδοσης συχνά διαθέτουν μεγαλύτερο εύρος παράλληλισμού λειτουργικών μονάδων από αυτό που μπορεί να χρησιμοποιήσει με αποτελεσματικό τρόπο κάποιο νήμα. Επιπλέον, με τη βοήθεια της μετονομασίας καταχωρητών και της δυναμικής χρονοδρομολόγησης, μπορούν να εκδίδονται πολλαπλές εντολές που ανήκουν σε ανεξάρτητα νήματα, χωρίς να χρειάζεται να λαμβάνεται μέριμνα για τις μεταξύ τους εξαρτήσεις. Ο χειρισμός της επίλυσης των εξαρτήσεων μπορεί να πραγματοποιηθεί μέσω των δυνατοτήτων της δυναμικής χρονοδρομολόγησης.

Το Σχήμα 3.8 απεικονίζει εννοιολογικά τις διαφορές όσον αφορά την ικανότητα ενός επεξεργαστή να αξιοποιεί τους πόρους των παρακάτω διατάξεων υπερβαθμισμένων επεξεργαστών:

- Υπερβαθμιστού επεξεργαστή χωρίς καμία υποστήριξη πολυνημάτωσης
- Υπερβαθμιστού επεξεργαστή με χονδρόκοκκη πολυνημάτωση
- Υπερβαθμιστού επεξεργαστή με λεπτόκοκκη πολυνημάτωση
- Υπερβαθμιστού επεξεργαστή με ταυτόχρονη πολυνημάτωση

Στον υπερβαθμιστό επεξεργαστή χωρίς καμία υποστήριξη πολυνημάτωσης, η χρήση των θυρίδων έκδοσης (issue slots) περιορίζεται από την έλλειψη του ILP, ζήτημα που εξετάστηκε στις προηγούμενες ενότητες. Επιπλέον, η εισαγωγή



Σχήμα 3.8 Τρόπος χρησιμοποίησης των θυρίδων έκδοσης ενός υπερβαθμωτού επεξεργαστή από τέσσερις διαφορετικές προσεγγίσεις. Ο οριζόντιος άξονας αναπαριστά τις δυνατότητες έκδοσης εντολών σε κάθε κύκλο ρολογιού. Ο κάθετος άξονας αναπαριστά την ακολουθία των κύκλων ρολογιού. Τα κενά (λευκά) πλαίσια υποδηλώνουν ότι η αντίστοιχη θυρίδα έκδοσης δεν χρησιμοποιείται σε αυτόν τον κύκλο ρολογιού. Τα σκιασμένα πλαίσια με αποχρώσεις του γαλάζιου και του μπλε αντιστοιχούν σε τέσσερα διαφορετικά νήματα που βρίσκονται στους πολυνηματικούς επεξεργαστές. Το μπλε χρώμα χρησιμοποιείται, επίσης, για την επισήμανση των κατειλημμένων θυρίδων έκδοσης στην περίπτωση του υπερβαθμωτού επεξεργαστή που δεν διαθέτει υποστήριξη πολυνημάτωσης. Ο επεξεργαστής Sun T1 (με το ακρωνύμιο Niagara), ο οποίος εξετάζεται στο επόμενο κεφάλαιο, αποτελεί παράδειγμα λεπτόκοκκης αρχιτεκτονικής.

ενός μεγάλου διαστήματος αδράνειας, όπως στην περίπτωση των αστοχιών της κρυφής μνήμης, μπορεί να θέσει σε αδράνεια ολόκληρο τον επεξεργαστή.

Στον υπερβαθμωτό επεξεργαστή χονδρόκοκκης πολυνημάτωσης, πραγματοποιείται εν μέρει απόκρυψη των διαστημάτων αδράνειας μεγάλης διάρκειας μέσω της αλλαγής σε κάποιο άλλο νήμα που χρησιμοποιεί τους πόρους του επεξεργαστή. Παρόλο που αυτό μειώνει τον αριθμό των εξ ολοκλήρου αδρανών κύκλων ρολογιού εντός κάθε κύκλου ρολογιού, οι περιορισμοί του ILP εξακολουθούν να οδηγούν σε αδρανείς κύκλους. Επιπλέον, στους επεξεργαστές χονδρόκοκκης πολυνημάτωσης, υπάρχει το ενδεχόμενο να παραμένουν μερικοί πλήρως αδρανείς κύκλοι, καθώς η αλλαγή των νημάτων προκύπτει μόνο όταν υπάρχει κάποιο διάστημα αδράνειας, ενώ συνάμα αξίζει να σημειωθεί ότι το νέο νήμα διαθέτει περίοδο εκκίνησης.

Στην περίπτωση της λεπτόκοκκης πολυνημάτωσης, η εναλλαγή των νημάτων εξαλείφει τις πλήρως κενές θυρίδες. Ωστόσο, επειδή μόνο ένα νήμα εκδίδει εντολές σε κάθε κύκλο ρολογιού, οι περιορισμοί του ILP εξακολουθούν να οδηγούν σε σημαντικό αριθμό αδρανών θυρίδων εντός των μεμονωμένων κύκλων ρολογιού.

Στην περίπτωση της SMT, πραγματοποιείται η ταυτόχρονη αξιοποίηση του

ILP και του TLP, όπου πολλαπλά νήματα χρησιμοποιούν τις θυρίδες έκδοσης εντός ενός μόνο κύκλου ρολογιού. Στην ιδανική περίπτωση, η χρήση των θυρίδων έκδοσης περιορίζεται από την ανισορροπία σε ό,τι αφορά τις ανάγκες που έχουν τα πολλαπλά νήματα για πόρους, καθώς και από τη διαθεσιμότητα των πόρων αυτών. Στην πράξη, υπάρχουν άλλοι παράγοντες - συμπεριλαμβανομένων του αριθμού των ενεργών νημάτων που εξετάζονται, των περιορισμών των απομονωτών πεπερασμένου μεγέθους, της ικανότητας ανάκτησης αρκετών εντολών από πολλαπλά νήματα και των πρακτικών περιορισμών που σχετίζονται με τους συνδυασμούς εντολών που μπορούν να εκδοθούν από ένα μόνο νήμα και από πολλαπλά νήματα - οι οποίοι μπορούν, επίσης, να περιορίσουν τον αριθμό των θυρίδων που χρησιμοποιούνται. Παρόλο που το Σχήμα 3.8 απλοποιεί σε σημαντικό βαθμό την πραγματική λειτουργία των επεξεργαστών αυτών, κατορθώνει να απεικονίζει τα ενδεχόμενα πλεονεκτήματα της πολυνημάτωσης στο επίπεδο της απόδοσης, τόσο σε γενικές γραμμές όσο και συγκεκριμένα για την SMT.

Όπως έχει ήδη αναφερθεί, η ταυτόχρονη πολυνημάτωση αξιοποιεί το γεγονός ότι οι δυναμικά χρονοδρομολογημένοι επεξεργαστές διαθέτουν πολλούς από τους μηχανισμούς υλικού που χρειάζονται για την υποστήριξη της ενοποιημένης υποστήριξης του TLP μέσω της πολυνημάτωσης. Συγκεκριμένα, οι δυναμικά χρονοδρομολογημένοι υπερβαθμωτοί επεξεργαστές διαθέτουν μεγάλο σύνολο εικονικών καταχωρητών, οι οποίοι μπορούν να χρησιμοποιηθούν για τη διατήρηση του συνόλου των καταχωρητών των ανεξάρτητων νημάτων (θεωρώντας ότι διατηρούνται ξεχωριστοί πίνακες μετονομασίας για κάθε νήμα). Επειδή η διαδικασία μετονομασίας των καταχωρητών παρέχει μοναδικούς προσδιοριστές καταχωρητών, οι εντολές που προέρχονται από πολλαπλά νήματα μπορούν να τοποθετούνται στη διαδρομή δεδομένων χωρίς να προκαλείται σύγχυση όσον αφορά τις αφειρήρες και τους προορισμούς των νημάτων.

Η παρατήρηση αυτή οδηγεί στο συμπέρασμα ότι η πολυνημάτωση μπορεί να εφαρμοστεί βασιζόμενη σε επεξεργαστή εκτός σειράς, προσθέτοντας έναν πίνακα μετονομασίας ανά νήμα, διατηρώντας ξεχωριστούς PCs και παρέχοντας τη δυνατότητα επικύρωσης σε εντολές πολλαπλών νημάτων.

Υπάρχουν επιπλοκές στο χειρισμό της επικύρωσης, καθώς επιθυμούμε να υπάρχει η δυνατότητα ανεξάρτητης επικύρωσης των εντολών που προέρχονται από ανεξάρτητα νήματα. Η ανεξάρτητη επικύρωση εντολών που προέρχονται από ξεχωριστά νήματα μπορεί να υποστηριχθεί διατηρώντας σε λογικό επίπεδο έναν ξεχωριστό απομονωτή αναδιάταξης για κάθε νήμα.

Προκλήσεις Σχεδίασης της SMT

Η SMT, σε περίπτωση που είναι χονδρόκοκκη, είναι απίθανο να οδηγήσει σε ιδιαίτερη βελτίωση της απόδοσης, εξαιτίας του γεγονότος ότι οι δυναμικά χρονοδρομολογημένοι υπερβαθμωτοί επεξεργαστές είναι πιθανό να διαθέτουν διασω-

λήνωση με μεγάλο βάθος. Αφού η SMT είναι λογικό να χρησιμοποιείται μόνο σε λεπτόκοκκες υλοποιήσεις, πρέπει να ανησυχούμε για την επίδραση που θα έχει η λεπτόκοκκη χρονοδρομολόγηση στην απόδοση του μεμονωμένου νήματος. Η επίδραση αυτή μπορεί να ελαχιστοποιηθεί έχοντας κάποιο προτιμώμενο νήμα (preferred thread), κάτι το οποίο εξακολουθεί να επιτρέπει στην πολυνημάτωση να διατηρεί μερικά από τα πλεονεκτήματα σε επίπεδο απόδοσης, κάνοντας ένα μικρό συμβιβασμό σε ό,τι αφορά την απόδοση του μεμονωμένου νήματος.

Εκ πρώτης όψεως, μπορεί να θεωρήσει κανείς ότι η προσέγγιση του προτιμώμενου νήματος δεν έχει επιπτώσεις ούτε στη ρυθμοαπόδοση ούτε στην απόδοση του μεμονωμένου νήματος. Δυστυχώς, στην προσέγγιση του προτιμώμενου νήματος ο επεξεργαστής είναι πιθανό να θυσιάζει σε κάποιο βαθμό τη ρυθμοαπόδοση, όταν το προτιμώμενο νήμα συναντά κάποιο διάστημα αδράνειας. Αιτία του παραπάνω φαινομένου είναι το γεγονός ότι η διασωλήνωση είναι λιγότερο πιθανό να περιλαμβάνει συνδυασμό εντολών που προέρχονται από αρκετά νήματα, γεγονός που οδηγεί σε μεγαλύτερη πιθανότητα εμφάνισης είτε κενών θυρίδων είτε διαστήματος αδράνειας. Η ρυθμοαπόδοση μεγιστοποιείται έχοντας επαρκή αριθμό ανεξάρτητων νημάτων, με στόχο την απόκρυψη όλων των διαστημάτων αδράνειας σε οποιονδήποτε συνδυασμό νημάτων.

Δυστυχώς, ο συνδυασμός πολλών νημάτων αναπόφευκτα επηρεάζει το χρόνο εκτέλεσης των μεμονωμένων νημάτων. Παρόμοια προβλήματα υφίστανται και στην ανάκτηση των εντολών. Προκειμένου να επιτευχθεί η μεγιστοποίηση της απόδοσης του μεμονωμένου νήματος, πρέπει να ανακτούμε εντολές από όσο το δυνατόν μεγαλύτερη απόσταση εντός του νήματος και πάντα να φροντίζουμε να είναι ελεύθερη η μονάδα ανάκτησης, όταν σημειώνεται εσφαλμένη πρόβλεψη διακλάδωσης και προκύπτει αστοχία στον απομονωτή προανάκτησης. Δυστυχώς, αυτό περιορίζει τον αριθμό των διαθέσιμων για χρονοδρομολόγηση εντολών από άλλα νήματα, γεγονός που οδηγεί σε μείωση της ρυθμοαπόδοσης. Όλοι οι πολυνηματικοί επεξεργαστές πρέπει να αναζητήσουν τρόπους εξισορρόπησης αυτής της αλληλοεξαρτώμενης σχέσης.

Στην πράξη, τα προβλήματα του διαμοιρασμού των πόρων και της εξισορρόπησης της απόδοσης του μεμονωμένου νήματος και των πολλαπλών νημάτων αποδεικνύονται όχι και τόσο απαιτητικά όσο ακούγονται, τουλάχιστον για τα σημερινά δεδομένα των υπερβαθμωτών επεξεργαστών. Συγκεκριμένα, στις σημερινές μηχανές που εκδίδουν τέσσερις έως οκτώ εντολές ανά κύκλο, είναι πιθανότατα αρκετό να έχουμε μικρό αριθμό ενεργών νημάτων και ακόμα μικρότερο αριθμό «προτιμώμενων» νημάτων. Ο επεξεργαστής, οποιοδήποτε αυτό είναι εφικτό, ενεργεί εκ μέρους κάποιου προτιμώμενου νήματος. Αυτό ξεκινά με την προανάκτηση των εντολών: οποιοδήποτε οι απομονωτές προανάκτησης των προτιμώμενων νημάτων δεν είναι πλήρεις, ανακτώνται εντολές για τα νήματα αυτά. Μόνο όταν οι απομονωτές προτιμώμενων νημάτων είναι πλήρεις προβαίνει η μονάδα εντολών στην προανάκτηση άλλων νημάτων. Αξίζει να σημειωθεί ότι η χρησιμοποίηση δύο προτιμώμενων νημάτων συνεπάγεται ότι πραγματοποιείται

ταυτόχρονη προανάκτηση δύο ροών εντολών, κάτι το οποίο αυξάνει την πολυπλοκότητα της μονάδας ανάκτησης εντολών και της κρυφής μνήμης εντολών. Παρόμοια, η μονάδα έκδοσης εντολών μπορεί να στρέψει την προσοχή της στα προτιμώμενα νήματα, εξετάζοντας τα υπόλοιπα νήματα μόνο σε περίπτωση που τα προτιμώμενα νήματα έχουν τεθεί σε αδράνεια και δεν μπορούν να εκδοθούν.

Υπάρχει πληθώρα επιπλέον προκλήσεων όσον αφορά τη σχεδίαση των επεξεργαστών SMT, συμπεριλαμβανομένων των ακόλουθων:

- Διαχείριση της χρήσης της μεγαλύτερης συστοιχίας καταχωρητών που απαιτείται για τη διατήρηση πολλαπλών περιβαλλόντων
- Διατήρηση του κύκλου ρολογιού στο ίδιο επίπεδο, ιδίως σε κρίσιμα βήματα, όπως αυτό της έκδοσης εντολών κατά το οποίο απαιτείται η εξέταση περισσότερων υποψήφιων εντολών και αυτό της ολοκλήρωσης των εντολών, όπου η επιλογή των εντολών που θα επικυρωθούν μπορεί να είναι απαιτητική διαδικασία
- Διασφάλιση ότι οι συγκρούσεις της κρυφής μνήμης και του TLB, που προκύπτουν εξαιτίας της ταυτόχρονης εκτέλεσης πολλαπλών νημάτων, δεν πρόκειται να οδηγήσουν σε σημαντική μείωση της απόδοσης

Παρατηρώντας τα προβλήματα αυτά, μπορεί κανείς να προβεί σε δύο σημαντικές παρατηρήσεις. Πρώτον, σε πολλές περιπτώσεις, η ενδεχόμενη επιβάρυνση της απόδοσης εξαιτίας της πολυνημάτωσης είναι μικρή και οι απλές επιλογές λειτουργούν ικανοποιητικά. Δεύτερον, η αποδοτικότητα των σημερινών υπερβαθμωτών επεξεργαστών είναι αρκετά χαμηλή, ώστε να μην υφίστανται αρκετά περιθώρια βελτιώσεων, ακόμα και αν αυτό θα σήμαινε σε κάποιο βαθμό επιπλέον επιβάρυνση.

Ο Power 5 της IBM χρησιμοποίησε την ίδια διασωλήνωση με αυτήν του Power 4, προσθέτοντας ωστόσο την υποστήριξη της SMT. Κατά την προσθήκη της SMT οι σχεδιαστές διαπίστωσαν ότι έπρεπε να αυξήσουν τον αριθμό των δομών του επεξεργαστή, έτσι ώστε να ελαχιστοποιηθούν οι επιπτώσεις της αρνητικής απόδοσης που προκύπτουν από την αλληλεπίδραση των λεπτόκοκκων νημάτων. Στις αλλαγές αυτές περιλαμβάνονταν τα ακόλουθα:

- Ενίσχυση του συσχετικού χαρακτήρα της κρυφής μνήμης εντολών L1 και των απομονωτών μετάφρασης διεύθυνσης εντολών
- Προσθήκη ουρών φόρτωσης και αποθήκευσης ανά νήμα
- Αύξηση του μεγέθους των κρυφών μνημών L2 και L3
- Προσθήκη ξεχωριστής προανάκτησης εντολών και αποθήκευσης σε απο-

- Αύξηση του αριθμού των εικονικών καταχωρητών από τους 152 στους 240
- Αύξηση του μεγέθους αρκετών ουρών έκδοσης

Η SMT, επειδή εκμεταλλεύεται τον παραλληλισμό επιπέδου νημάτων στους υπερβαθμωτούς επεξεργαστές πολλαπλής έκδοσης, είναι πιο πιθανό να συμπεριληφθεί στους επεξεργαστές υψηλής απόδοσης που απευθύνονται στην αγορά των εξυπηρετητών. Επιπλέον, είναι πιθανή η ύπαρξη κάποιας κατάστασης λειτουργίας, κατά την οποία θα περιορίζεται η πολυνημάτωση, έτσι ώστε να μεγιστοποιείται η απόδοση του μεμονωμένου νήματος.

Ευδεχόμενα Πλεονεκτήματα σε Επίπεδο Απόδοσης της SMT

Το βασικό ερώτημα είναι: «Σε ποιο βαθμό μπορεί να αυξηθεί η απόδοση μέσω της υλοποίησης της SMT;». Οι ερευνητές που επιχείρησαν να απαντήσουν στο ερώτημα αυτό κατά την περίοδο 2000-2001 υπέθεσαν ότι οι δυναμικοί υπερβαθμωτοί επεξεργαστές θα αποκτούσαν ιδιαίτερα υψηλότερο εύρος στα επόμενα πέντε έτη, παρέχοντας υποστήριξη έκδοσης έξι έως οκτώ εντολών ανά κύκλο μέσω της εφαρμογής της υποθετικής δυναμικής χρονοδρομολόγησης, πολλών ταυτόχρονων εντολών φόρτωσης και αποθήκευσης, μεγάλων κύριων κρυφών μνημών και τεσσάρων έως οκτώ περιβαλλόντων με παροχή δυνατότητας ταυτόχρονης ανάκτησης από πολλαπλά περιβάλλοντα. Για διάφορους λόγους, οι οποίοι θα αποσαφηνιστούν στην επόμενη ενότητα, κανένας επεξεργαστής με αυτές τις δυνατότητες δεν έχει μέχρι σήμερα αναπτυχθεί και ούτε πρόκειται να κατασκευαστεί στο κοντινό μέλλον.

Φυσικό επακόλουθο του παραπάνω φαινομένου είναι το γεγονός ότι τα προσομοιωμένα ερευνητικά αποτελέσματα, τα οποία προέβλεπαν το διπλασιασμό ή τον πολλαπλασιασμό των ωφελειών που θα προέκυπταν για τα πολυπρογραμματιζόμενα φορτία εργασιών, δεν είναι ρεαλιστικά. Στην πράξη, οι υφιστάμενες υλοποιήσεις της SMT παρέχουν μόνο δύο περιβάλλοντα όπου η ανάκτηση πραγματοποιείται μόνο από το ένα από αυτά, καθώς και δυνατότητες έκδοσης χαμηλότερου βεληνεκούς. Αποτέλεσμα αυτού είναι το γεγονός ότι τα οφέλη που προκύπτουν από την SMT είναι, επίσης, χαμηλότερου βεληνεκούς.

Για παράδειγμα, στον Pentium 4 Extreme, όπως αυτός υλοποιήθηκε στους εξυπηρετητές της HP-Compaq, η χρήση της SMT, κατά την εκτέλεση του μετροπρογράμματος SPECintRate, είχε ως αποτέλεσμα τη βελτίωση της απόδοσης κατά 1.01, ενώ, κατά την εκτέλεση του μετροπρογράμματος SPECfpRate, είχε ως αποτέλεσμα τη βελτίωση της απόδοσης κατά 1.07. Σε ξεχωριστή μελέτη, οι Tuck και Tullsen [2003] παρατήρησαν ότι η εκτέλεση του συνδυασμού καθενός από τα 26 μετροπρογράμματα του SPEC με κάθε μετροπρόγραμμα του SPEC (εκτέλεση, δηλαδή, 26² ζευγών μετροπρογραμμάτων, με την προϋπόθεση ότι κάθε μετροπρόγραμμα εκτελούνταν σε συνδυασμό και με τον εαυτό του) είχε

ως αποτέλεσμα επιτάχυνση (speedup) που κυμαινόταν ανάμεσα στο 0.90 και το 1.58, ενώ η μέση επιτάχυνση ισούτο με 1.20 (αξίζει να σημειωθεί ότι οι μετρήσεις αυτές είναι διαφορετικές από τη SPECRate, η οποία απαιτεί την εκτέλεση καθενός από τα μετροπρογράμματα του SPEC σε συνδυασμό με πλήθος αντιγράφων του ίδιου μετροπρογράμματος που επιλέγεται από τον προμηθευτή). Στα παράλληλα μετροπρογράμματα του Splash, οι ίδιοι ερευνητές ανέφεραν επιτάχυνση που κυμαίνεται ανάμεσα στο 1.02 και το 1.67, ενώ η μέση επιτάχυνση ισούτο περίπου με 1.22.

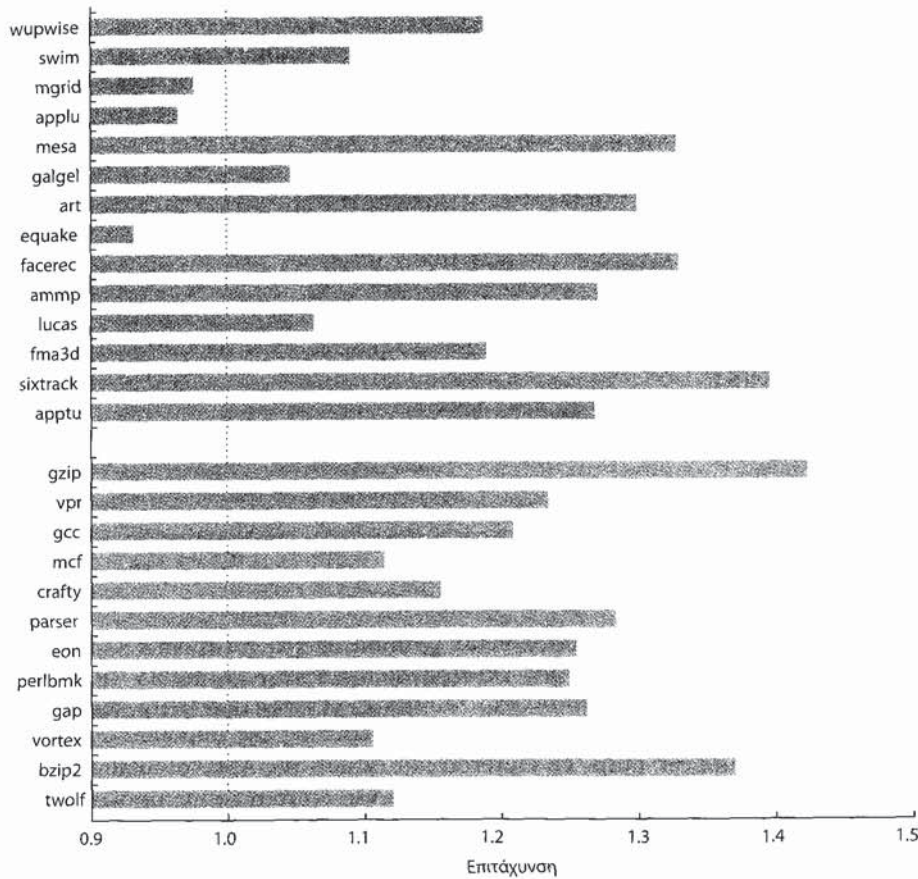
Ο Power 5 της IBM αποτελεί την πιο εντατική υλοποίηση της SMT έως το 2005 και διαθέτει εκτεταμένες προσθήκες υποστήριξης της, όπως αυτές περιγράφηκαν στην προηγούμενη υποενότητα. Η άμεση σύγκριση της απόδοσης του Power 5 σε κατάσταση λειτουργίας SMT, εκτελώντας στον επεξεργαστή δύο αντίγραφα μίας εφαρμογής, με τον Power 5 σε κατάσταση λειτουργίας ενός μόνο νήματος, έχοντας μία μόνο διεργασία ανά πυρήνα, παρουσιάζει, για πληθώρα μετροπρογραμμάτων, επιτάχυνση που κυμαίνεται μεταξύ του 0.89 (απώλεια απόδοσης) και του 1.41. Παρ' όλα αυτά, οι περισσότερες εφαρμογές αποκόμισαν κάποιο όφελος από την SMT. Οι εντατικές εφαρμογές κινητής υποδιαστολής, στις οποίες σημειώθηκαν οι περισσότερες συγκρούσεις της κρυφής μνήμης, παρουσίασαν τα λιγότερα οφέλη.

Το Σχήμα 3.9 παρουσιάζει την επιτάχυνση του πολυεπεξεργαστή Power 5 με τους 8 επεξεργαστές με ή χωρίς την εφαρμογή της SMT για τα μετροπρογράμματα του SPECRate2000, όπως αυτή περιγράφεται στη λεζάντα του σχήματος. Κατά μέσο όρο, το SPECRate2000 είναι 1.23 φορές ταχύτερο, ενώ το SPECfpRate είναι 1.16 φορές ταχύτερο. Αξίζει να σημειωθεί ότι μερικά μετροπρογράμματα κινητής υποδιαστολής σημειώνουν μια μικρή μείωση της απόδοσης σε κατάσταση λειτουργίας SMT, με τη μέγιστη μείωση της απόδοσης να ανέρχεται σε 0.93.

Τα αποτελέσματα αυτά είναι προφανές ότι καταδεικνύουν τα οφέλη που προκύπτουν από την SMT για τους εντατικούς υποθετικούς επεξεργαστές που διαθέτουν εκτεταμένη υποστήριξη της SMT. Ωστόσο, εξαιτίας του κόστους και της φθίνουσας ανταποδοτικής ωφέλειας της SMT, πολλοί σχεδιαστές επιλέγουν, αντί να υλοποιούν υπερβαθμωτούς επεξεργαστές με υψηλότερο εύρος και πιο εντατικές εκδόσεις της SMT, να ενσωματώνουν πολλαπλούς πυρήνες CPU σε ένα μόνο κύβο, έχοντας ελαφρά λιγότερο εντατική υποστήριξη της πολλαπλής έκδοσης και της πολυνημάτωσης. Θα επιστρέψουμε το ζήτημα αυτό στο επόμενο κεφάλαιο.

3.6 Συνδυασμός Όλων των Δεδομένων: Απόδοση και Αποδοτικότητα των Προηγμένων Επεξεργαστών Πολλαπλής Έκδοσης

Στην ενότητα αυτή διερευνούμε τα χαρακτηριστικά αρκετών πρόσφατων επεξεργαστών πολλαπλής έκδοσης και εξετάζουμε την απόδοση και την αποδοτικότητά



Σχήμα 3.9 Σύγκριση της απόδοσης του IBM eServer p5 575 των 8 επεξεργαστών σε κατάσταση λειτουργίας SMT και σε κατάσταση λειτουργίας ενός μόνο νήματος (single-thread, ST). Αξίζει να σημειωθεί ότι ο άξονας y ξεκινά με επιτάχυνση της τάξης του 0.9, που ισοδυναμεί με απώλεια απόδοσης. Σε κάθε πυρήνα του Power 5 είναι ενεργός μόνο ένας επεξεργαστής, κάτι το οποίο βελτιώνει ελαφρά τα αποτελέσματα της SMT μέσω της μείωσης των καταστροφικών παρεμβολών στο σύστημα μνήμης. Τα αποτελέσματα της SMT έχουν αποκτηθεί μέσω της δημιουργίας 16 νημάτων χρήστη, ενώ τα αποτελέσματα του ST χρησιμοποιούν μόνο 8 νήματα. Έχοντας μόνο ένα νήμα ανά επεξεργαστή, ο Power 5 μεταφέρεται σε κατάσταση λειτουργίας ενός μόνο νήματος από το λειτουργικό σύστημα. Τα αποτελέσματα αυτά συλλέχθηκαν από τον John McCalpin της IBM. Όπως μπορεί να διαπιστώσει κανείς από τα δεδομένα, η τυπική απόκλιση των αποτελεσμάτων του SPECfpRate είναι υψηλότερη από αυτήν του SPECintRate (0.13 έναντι 0.07), κάτι το οποίο υποδηλώνει ότι η βελτίωση που υφίστανται τα προγράμματα FP από την SMT είναι πιθανό να διαφέρει ανά περίπτωση σε μεγάλο βαθμό.

(efficiency) τους όσον αφορά τη χρήση του πυριτίου, των transistors και της ενέργειας. Στη συνέχεια μελετώνται τα πρακτικά όρια των υπερβαθμωτών επε-

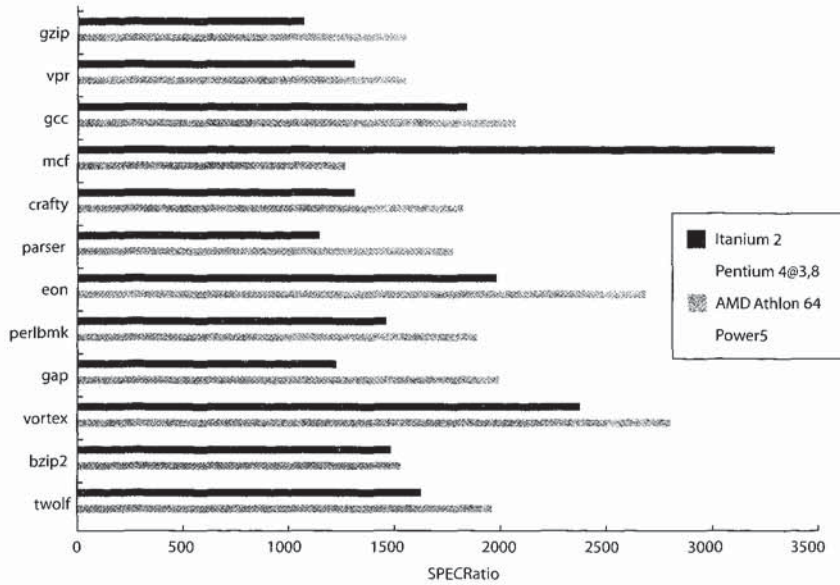
Επεξεργαστής	Μικροαρχιτεκτονική	Ανάκτηση/ έκδοση/ εκτέλεση	Λειτ. μονάδες	Ρυθμός ρολογιού (GHz)	Transistors και μέγεθος κύβου	Ισχύς
Intel Pentium 4 Extreme	Υποθετική δυναμικής χρονοδρομολόγησης. Διασωλήνωση μεγάλου βάθους. SMT	3/3/4	7 ακ. 1 FP	3.8	125M 122 mm ²	115 W
AMD Athlon 64 FX-57	Υποθετικής δυναμικής χρονοδρομολόγησης	3/3/4	6 ακ. 3 FP	2.8	114M 115 mm ²	104 W
IBM Power5 ενός επεξεργαστή	Υποθετικής δυναμικής χρονοδρομολόγησης. SMT. Δύο πυρήνων/chip	8/4/8	6 ακ. 2 FP	1.9	200M 300 mm ² (εκτιμώμενα μεγέθη)	80 W (εκτιμώμενα μεγέθη)
Intel Itanium 2	Τεχνοτροπίας EPIC. Στατικής κυρίως χρονοδρομολόγησης	6/5/11	9 ακ. 2 FP	1.6	592M 423 mm ²	130 W

Σχήμα 3.10 Τα χαρακτηριστικά τεσσάρων πρόσφατων επεξεργαστών πολλαπλής έκδοσης. Ο Power 5 διαθέτει δύο πυρήνες CPU, παρόλο που στο κεφάλαιο αυτό εξετάζουμε την απόδοση του ενός πυρήνα. Ο αριθμός των transistors, το εμβαδόν και η κατανάλωση της ισχύος του ενός πυρήνα του Power 5 έχουν υπολογιστεί με βάση τις μετρήσεις που έχουν πραγματοποιηθεί για τους δύο πυρήνες και έχουν τιμές ίσες με 276M, 389 mm² και 125 W αντιστοίχως. Ο κύβος μεγάλου μεγέθους, καθώς και ο υψηλός αριθμός των transistors εν μέρει εξηγείται εξαιτίας της παρουσίας και τρίτης κρυφής μνήμης των 9M στο chip. Οι Opteron και Athlon της AMD διαθέτουν την ίδια μικροαρχιτεκτονική πυρήνα. Ο Athlon προορίζεται για επιτραπέζιους υπολογιστές και δεν υποστηρίζει την πολυεπεξεργασία. Ο Opteron προορίζεται για εξυπηρετητές και υποστηρίζει την πολυεπεξεργασία. Είναι παρόμοια περίπτωση με αυτή των επεξεργαστών Pentium και Xeon της σειράς προϊόντων της Intel.

ξεργαστών και το μέλλον των μικροεπεξεργαστών υψηλής απόδοσης.

Το Σχήμα 3.10 παρουσιάζει τα χαρακτηριστικά τεσσάρων από τους πρόσφατους μικροεπεξεργαστές υψηλής απόδοσης. Διαφέρουν σε σημαντικό βαθμό όσον αφορά την οργάνωση, το ρυθμό έκδοσης, τις δυνατότητες των λειτουργικών μονάδων, το ρυθμό του ρολογιού, το μέγεθος κύβου, τον αριθμό των transistors και την ισχύ τους. Όπως μπορεί να διαπιστώσει κανείς εξετάζοντας τα Σχήματα 3.11 και 3.12, δεν υπάρχει κανένας μικροεπεξεργαστής που να προηγείται εμφανώς σε ό,τι αφορά την απόδοση. Οι Itanium 2 και Power 5, οι οποίοι παρουσιάζουν παρόμοια απόδοση στο SPECfp, υπερέχουν εμφανώς στα μετροπρογράμματα αυτά σε σύγκριση με τους Athlon και Pentium 4. Ο Athlon της AMD διαθέτει την υψηλότερη απόδοση στα μετροπρογράμματα SPECint και στη συνέχεια ακολουθούν οι Pentium 4, Itanium 2 και Power 5.

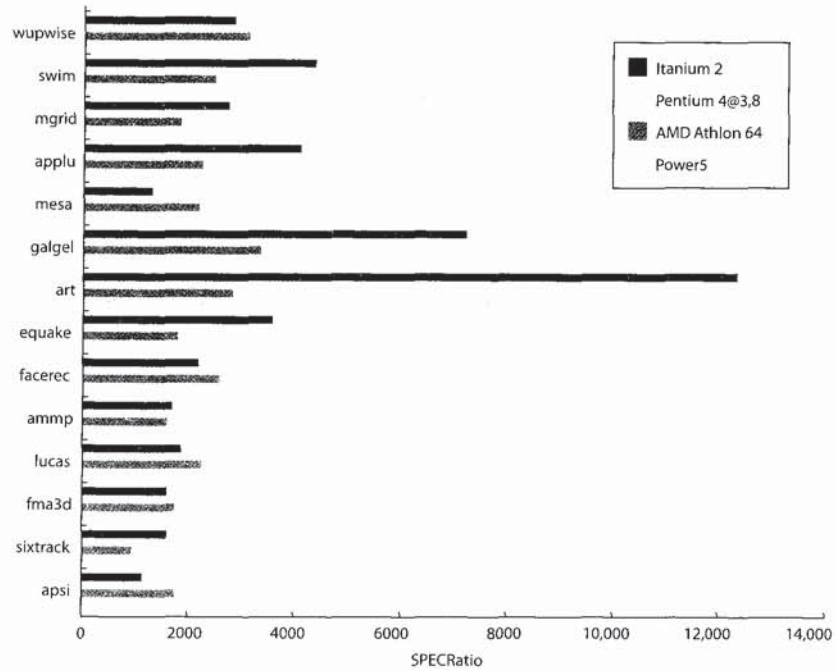
Το ερώτημα που σχετίζεται με την αποδοτικότητα του εμβαδού του πυριτίου και της ισχύος έχει παρόμοια σπουδαιότητα με αυτό της συνολικής απόδοσης. Όπως διαπιστώσαμε στο Κεφάλαιο 1, η ισχύς έχει καταστεί ο βασικός περιο-



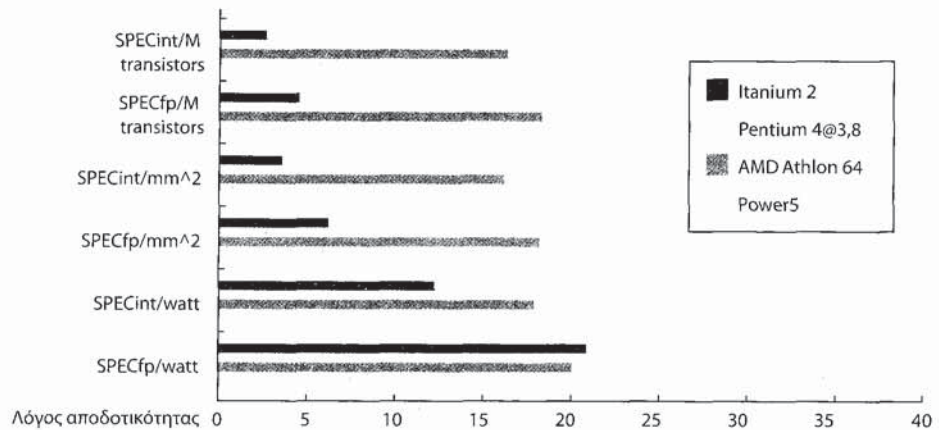
Σχήμα 3.11 Σύγκριση της απόδοσης των τεσσάρων προηγμένων επεξεργαστών πολλαπλής έκδοσης, που παρουσιάζονται στο Σχήμα 3.10, όσον αφορά τα μετροπρογράμματα του SPECint2000.

ρισμός των σύγχρονων επεξεργαστών. Στο Σχήμα 3.13 παρουσιάζεται ο τρόπος σύγκρισης των επεξεργαστών αυτών σε ό,τι αφορά την αποδοτικότητά τους, μέσω της διαγραμματικής απεικόνισης της απόδοσης του SPECint του SPECfp σε σύγκριση με τον αριθμό των transistors, το εμβαδόν του πυριτίου και την ισχύ. Ενδιαφέρον προκαλεί το γεγονός ότι αποτελέσματα αυτά είναι αντίθετα από τα αποτελέσματα της απόδοσης. Ο Itanium 2 είναι πιο αποδοτικός επεξεργαστής τόσο σε ό,τι αφορά τον κώδικα κινητής υποδιαστολής όσο και αναφορικά με τον κώδικα ακεραίων για όλες τις μετρήσεις εκτός από μία (SPECfp/watt). Σε ό,τι αφορά την αποδοτικότητα, τόσο ο Athlon όσο και ο Pentium 4 προβαίνουν σε ορθή χρήση των transistors και του εμβαδού, ενώ ο Power 5 της IBM πραγματοποιεί την πιο αποδοτική χρήση της ενέργειας για το SPECfp, την ίδια ώρα που στο SPECint υπάρχει ουσιαστικά ισοδυναμία των διάφορων αρχιτεκτονικών. Το γεγονός ότι κανένας από τους επεξεργαστές δεν διαθέτει μεγάλη διαφορά σε επίπεδο αποδοτικότητας μας οδηγεί στο συμπέρασμα ότι καμία προσέγγιση δεν αποτελεί πανάκεια που θα επιτρέψει τη ραγδαία αύξηση, σε σύγκριση με τα σημερινά δεδομένα, του βαθμού αξιοποίησης του ILP με εύκολο και αποδοτικό τρόπο.

Ας επιχειρήσουμε να διαπιστώσουμε γιατί ισχύουν τα παραπάνω.



Σχήμα 3.12 Σύγκριση της απόδοσης των τεσσάρων προηγμένων επεξεργαστών πολλαπλής έκδοσης, που παρουσιάζονται στο Σχήμα 3.10, όσον αφορά τα μετροπρογράμματα του SPECfp2000.



Σχήμα 3.13 Μετρήσεις της αποδοτικότητας των τεσσάρων επεξεργαστών πολλαπλής έκδοσης. Στην περίπτωση του Power 5, ο ένας μόνο κύβος περιλαμβάνει δύο πυρήνες επεξεργαστών και ως εκ τούτου υπολογίζουμε ότι οι αντίστοιχες μετρήσεις για τον ένα πυρήνα είναι ίσες με 80 W για την ισχύ, με 290 mm² για το εμβαδόν και με 200M για τον αριθμό των transistors.

Τι Είναι Αυτό που Περιορίζει τους Επεξεργαστές Πολλαπλής Έκδοσης :

Οι περιορισμοί που εξετάστηκαν στις Ενότητες 3.1 και 3.3 λειτουργούν ως σημαντικά εμπόδια για την περαιτέρω αξιοποίηση του ILP, ωστόσο δεν αποτελούν τα μόνα εμπόδια που υφίστανται. Για παράδειγμα, ο διπλασιασμός του ρυθμού έκδοσης από τα σημερινά επίπεδα των 3-6 εντολών ανά κύκλο ρολογιού, ας πούμε, σε 6-12 εντολές, πιθανότατα απαιτεί να είναι σε θέση ο επεξεργαστής να προβαίνει στην έκδοση τριών ή τεσσάρων προσπελάσεων της μνήμης ανά κύκλο, στην επίλυση τριών διακλαδώσεων ανά κύκλο, στη μετονομασία και προσπέλαση περισσότερων από 20 καταχωρητών ανά κύκλο και στην ανάκτηση 12-14 εντολών ανά κύκλο. Η πολυπλοκότητα υλοποίησης των δυνατοτήτων αυτών είναι πιθανό να συνεπάγεται την πραγματοποίηση θυσιών σε ό,τι αφορά το μέγιστο ρυθμό ρολογιού. Για παράδειγμα, ο επεξεργαστής του Σχήματος 3.10 με το μεγαλύτερο εύρος έκδοσης είναι ο Itanium 2, ο οποίος, ωστόσο, διαθέτει το χαμηλότερο ρυθμό ρολογιού, παρά το γεγονός ότι καταναλώνει την περισσότερη ισχύ.

Είναι πλέον ευρέως αποδεκτό ότι οι σύγχρονοι μικροεπεξεργαστές διαθέτουν κυρίως περιορισμούς όσον αφορά την ισχύ. Η ισχύς είναι συνάρτηση τόσο της στατικής ισχύος, η οποία αυξάνεται κατ' αναλογία του αριθμού των transistors (ανεξαρτήτως του αν τα transistors μετάγονται ή όχι) όσο και της δυναμικής ισχύος, η οποία είναι ανάλογη του γινομένου του αριθμού των transistors που μετάγονται και του ρυθμού μεταγωγής. Παρόλο που η στατική ισχύς αποτελεί, όπως είναι φυσικό, ζήτημα που πρέπει να λαμβάνεται υπόψη, κατά τη διάρκεια λειτουργίας η δυναμική ισχύς αποτελεί τον πρωτεύοντα παράγοντα κατανάλωσης ενέργειας. Οι μικροεπεξεργαστές που επιθυμούν να διαθέτουν ταυτόχρονα χαμηλό CPI και υψηλό CR πρέπει να πραγματοποιούν τη μεταγωγή περισσότερων transistors με ταχύτερο ρυθμό, γεγονός που οδηγεί στην αύξηση της κατανάλωσης ισχύος η οποία αποτελεί γινόμενο δύο αυτών στοιχείων.

Βεβαίως οι περισσότερες τεχνικές αύξησης της απόδοσης, συμπεριλαμβανομένων της χρήσης πολλαπλών πυρήνων και της πολυνημάτωσης, οδηγούν σε αύξηση της κατανάλωσης ισχύος. Το βασικό ερώτημα αφορά το αν κάποια τεχνική είναι *αποδοτική σε επίπεδο ενέργειας*: Αυξάνει η τεχνική την κατανάλωση ισχύος με ταχύτερο ρυθμό από αυτόν με τον οποίο βελτιώνει την απόδοση; Δυστυχώς, όλες οι τεχνικές που σήμερα χρησιμοποιούμε για την ενίσχυση της απόδοσης είναι σε αυτό το επίπεδο μη αποδοτικές, κάτι το οποίο προκύπτει από δύο βασικά χαρακτηριστικά.

Πρώτον, η έκδοση πολλαπλών εντολών επιφέρει κάποια επιβάρυνση στη λογική, η οποία αυξάνεται ταχύτερα από το ρυθμό αύξησης της έκδοσης. Η λογική αυτή είναι υπεύθυνη για την ανάλυση της έκδοσης εντολών, συμπεριλαμβανομένου του ελέγχου των εξαρτήσεων, της μετονομασίας των καταχωρητών και παρόμοιων λειτουργιών. Το τελικό αποτέλεσμα είναι ότι, χωρίς να έχουμε μειώ-

σεις της τάσης για τη μείωση της ισχύος, όσο χαμηλότερο είναι το CPI τόσο χαμηλότερος είναι ο λόγος απόδοσης ανά watt, εξαιτίας της επιβάρυνσης.

Δεύτερον και πιο σημαντικό είναι η αύξηση του χάσματος ανάμεσα στο μέγιστο ρυθμό έκδοσης και τη μέση απόδοση. Καθώς ο αριθμός των transistors που μεταγόνται είναι ανάλογος του μέγιστου ρυθμού έκδοσης και η απόδοση είναι ανάλογη του μέσου ρυθμού, η αύξηση του χάσματος ανάμεσα στη μέγιστη και τη μέση απόδοση μεταφράζεται σε αύξηση της ενέργειας ανά μονάδα απόδοσης. Δυστυχώς, το αυξανόμενο αυτό χάσμα φαίνεται πως έχει θεμελιώδη σημασία και προκύπτει εξαιτίας πολλών από τα ζητήματα που εξετάσαμε στις Ενότητες 3.2 και 3.3. Για παράδειγμα, αν θέλουμε να *διατηρούμε* τέσσερις εντολές ανά κύκλο, πρέπει να *ανακτούμε*, να *εκδίδουμε* και να *εκκινούμε την εκτέλεση* περισσότερων από τεσσάρων εντολών. Η ισχύς θα είναι ανάλογη του μέγιστου ρυθμού, ωστόσο η απόδοση θα διατηρείται στο επίπεδο του μέσου ρυθμού. (Σε πολλούς πρόσφατους επεξεργαστές, έχει ληφθεί μέριμνα για τη μείωση της κατανάλωσης ισχύος, μέσω του τερματισμού της λειτουργίας των ανενεργών τμημάτων του επεξεργαστή, συμπεριλαμβανομένης και της απενεργοποίησης του ρολογιού για αυτό το τμήμα του επεξεργαστή. Τέτοιες τεχνικές, παρόλο που είναι χρήσιμες, δεν μπορούν να παρεμποδίσουν τη μακροχρόνια μείωση της αποδοτικότητας της ισχύος.)

Επιπλέον, η πιο σημαντική τεχνική της τελευταίας δεκαετίας για την αύξηση του βαθμού αξιοποίησης του ILP - γνωστή ως εικασία - είναι από τη φύση της μη αποδοτική. Γιατί; Διότι δεν μπορεί να είναι ποτέ απόλυτα ακριβής. Δηλαδή, η εικασία από τη φύση της οδηγεί στην εκτέλεση περιττών υπολογισμών, πριν να γνωρίζουμε αν αυτοί έχουν ως αποτέλεσμα την πρόοδο του προγράμματος.

Αν η εικασία ήταν απόλυτα ακριβής, θα μπορούσε να εξοικονομεί ενέργεια, καθώς θα οδηγούσε στη μείωση του χρόνου εκτέλεσης και στην εξοικονόμηση στατικής ισχύος, ενώ συγχρόνως θα προσέθετε, σε κάποιο βαθμό, επιπλέον επιβάρυνση υλοποίησης. Η εικασία, όταν δεν είναι απόλυτα ακριβής, καθίσταται ταχύτατα μη αποδοτική σε επίπεδο ενέργειας, καθώς απαιτεί επιπλέον δυναμική ισχύ τόσο για την εσφαλμένη εικασία όσο και για την εκκαθάριση της κατάστασης του επεξεργαστή. Εξαιτίας της επιβάρυνσης που προκύπτει από την υλοποίηση της εικασίας - από τη μετονομασία καταχωρητών, τους απομονωτές αναδιάταξης, το μεγαλύτερο αριθμό απομονωτών κ.ο.κ. - δεν είναι ιδιαίτερα πιθανό το ενδεχόμενο οι υποθετικοί επεξεργαστές να είναι σε θέση να εξοικονομούν ενέργεια για μεγάλο εύρος ρεαλιστικών προγραμμάτων.

Τι ισχύει όμως στην περίπτωση της προσπάθειας βελτίωσης του ρυθμού του ρολογιού; Δυστυχώς και στην περίπτωση προσπάθειας αύξησης του ρυθμού ρολογιού ισχύει ένας παρόμοιος γρίφος: η αύξηση του ρυθμού του ρολογιού οδηγεί στην αύξηση της συχνότητας μεταγωγής των transistors, καθώς και στην άμεση αύξηση της κατανάλωσης της ισχύος. Για την επίτευξη υψηλότερου ρυθμού ρολογιού, χρειάζεται η αύξηση του βάθους της διασωλήνωσης. Ωστόσο, οι διασωλήνώσεις που έχουν μεγαλύτερο βάθος διαθέτουν επιπρόσθετες ποινές

επιβάρυνσης, ενώ συγχρόνως προκαλούν υψηλότερο ρυθμό μεταγωγής.

Το καλύτερο παράδειγμα αναπαράστασης του φαινομένου αυτού αποτελεί η σύγκριση του Pentium III με τον Pentium 4. Σε πρώτη ανάγνωση, ο Pentium 4 αποτελεί έκδοση της αρχιτεκτονικής του Pentium III με μεγαλύτερο βάθος διασωλήνωσης. Με παρόμοιο τρόπο, ο Pentium 4 καταναλώνει περισσότερη ισχύ, η οποία είναι ανάλογη της διαφοράς που υφίσταται στο ρυθμό του ρολογιού. Δυστυχώς, η απόδοση του Pentium 4 είναι μικρότερη από το λόγο του ρυθμού των ρολογιών των δύο επεξεργαστών, εξαιτίας επιπλέον επιβάρυνσης και περιορισμών του ILP.

Φαίνεται πως, στην προσπάθειά μας να αξιοποιήσουμε τον ILP, έχουμε φτάσει - και, σε μερικές περιπτώσεις, έχουμε ξεπεράσει - το όριο της φθίνουσας απόδοσης. Τα αποτελέσματα του γεγονότος αυτού είναι ευδιάκριτα αν παρατηρήσει κανείς τα τελευταία έτη το χαμηλότερο ρυθμό αύξησης της απόδοσης (δείτε το Κεφάλαιο 1), την έλλειψη βελτίωσης των δυνατοτήτων έκδοσης και την εμφάνιση των σχεδιάσεων με πολλαπλούς πυρήνες. Θα επιστρέψουμε στο ζήτημα αυτό στις συμπερασματικές παρατηρήσεις.

3.7 Πλάνες και Παγίδες

Πλάνη: Υπάρχει μια απλή προσέγγιση που σχετίζεται με τους επεξεργαστές πολλαπλής έκδοσης, η οποία έχει ως αποτέλεσμα την επίτευξη υψηλής απόδοσης, χωρίς να υφίσταται σημαντική επένδυση στο εμβαδόν του πυριτίου ή στην πολυπλοκότητα της σχεδίασης.

Οι τελευταίες ενότητες πρέπει να το έχουν καταστήσει αυτό σαφές. Αυτό που έχει αποτελέσει έκπληξη είναι το γεγονός ότι πολλοί σχεδιαστές θεωρούσαν ότι η άποψη αυτή ήταν ορθή και κατέβαλλαν σημαντικές προσπάθειες εύρεσης της βέλτιστης προσέγγισης. Παρόλο που είναι δυνατό να δημιουργήσουμε σχετικά απλούς επεξεργαστές πολλαπλής έκδοσης, όσο αυξάνεται ο ρυθμός έκδοσης, εμφανίζεται ο νόμος της φθίνουσας απόδοσης, ενώ το κόστος του πυριτίου και ενέργειας της έκδοσης μεγαλύτερου εύρους είναι αυτό που υπερφαλαγγίζει τα οφέλη που προκύπτουν σε επίπεδο απόδοσης.

Εκτός από τη μη αποδοτικότητα του υλικού, είναι σαφές ότι, στους επεξεργαστές που διαθέτουν σημαντικό βαθμό ILP, η μεταγλώττιση έχει καταστεί ιδιαίτερα πολύπλοκη. Όχι μόνο χρειάζεται ο μεταγλωττιστής να υποστηρίξει μεγάλο αριθμό εξεζητημένων μετασχηματισμών, αλλά παράλληλα η ρύθμιση του μεταγλωττιστή για την επίτευξη υψηλής απόδοσης για μεγάλο αριθμό μετροπρογραμμάτων φαίνεται ότι είναι πολύ δύσκολη.

Η επίτευξη υψηλής απόδοσης επηρεάζεται, επίσης, από αποφάσεις σχεδίασης σε επίπεδο συστήματος και αυτού του είδους οι επιλογές μπορεί να είναι ιδιαίτερα πολύπλοκες, όπως φάνηκε ξεκάθαρα στην τελευταία ενότητα.

Παγίδα: Η βελτίωση μόνο ενός χαρακτηριστικού των επεξεργαστών πολλαπλής έκδοσης συνεπάγεται τη συνολική βελτίωση της απόδοσης.

Η παγίδα αυτή αποτελεί απλά επαναδιατύπωση του νόμου του Amdahl. Ο σχεδιαστής μπορεί απλά να εξετάσει κάποια σχεδίαση, να διαπιστώσει ότι υφίσταται απλός μηχανισμός πρόβλεψης διακλάδωσης και να τον βελτιώσει, αναμένοντας τη σημαντική βελτίωση της απόδοσης. Η δυσκολία έγκειται στο γεγονός ότι πολλοί είναι οι παράγοντες που περιορίζουν την απόδοση των μηχανών πολλαπλής έκδοσης και η βελτίωση μόνο ενός χαρακτηριστικού του επεξεργαστή συχνά έχει ως αποτέλεσμα την εμφάνιση ζητημάτων που προηγουμένως δεν περιόριζαν την απόδοση.

Σχετικά παραδείγματα μπορεί να διακρίνει κανείς παρατηρώντας τα δεδομένα που σχετίζονται με τον ILP. Για παράδειγμα, εξετάζοντας απλά τις επιπτώσεις της πρόβλεψης διακλάδωσης στο Σχήμα 3.3 στη σελίδα 219, μπορούμε να διαπιστώσουμε ότι η μεταφορά από τον τυποποιημένο μηχανισμό πρόβλεψης των 2 bits στον επιλεκτικό μηχανισμό πρόβλεψης βελτιώνει άμεσα σε σημαντικό βαθμό τον παραλληλισμό (από ρυθμό έκδοσης της τάξης του 7 σε ρυθμό έκδοσης της τάξης του 12). Ωστόσο, αν ο επεξεργαστής παρέχει μόνο 32 καταχωρητές για μετονομασία, το εύρος του παραλληλισμού περιορίζεται στην έκδοση 5 εντολών ανά κύκλο ρολογιού, ακόμα και αν η προσέγγιση της πρόβλεψης διακλάδωσης είναι καλύτερη από οποιαδήποτε εναλλακτική λύση.

3.8 Συμπερασματικές Παρατηρήσεις

Τα σχετικά οφέλη των προσεγγίσεων εντατικής χρήσης λογισμικού και εντατικής χρήσης υλικού για την αξιοποίηση του ILP εξακολουθούν να αποτελούν αντικείμενο συζήτησης, παρόλο που η συζήτηση αυτή έχει αλλάξει κατεύθυνση τα τελευταία πέντε έτη. Αρχικά, οι προσεγγίσεις εντατικής χρήσης λογισμικού και εντατικής χρήσης υλικού ήταν διαφορετικές, ενώ αμφίβολη ήταν η δυνατότητα διαχείρισης της πολυπλοκότητας των προσεγγίσεων εντατικής χρήσης υλικού. Η ανάπτυξη αρκετών υποθετικών δυναμικών επεξεργαστών υψηλής απόδοσης, οι οποίοι διαθέτουν υψηλό ρυθμό ρολογιού, άμβλυσε τις ανησυχίες αυτές.

Η πολυπλοκότητα της αρχιτεκτονικής IA-64 και της σχεδίασης του Itanium αποτέλεσε για πολλούς σχεδιαστές την ένδειξη ότι οι προσεγγίσεις εντατικής χρήσης του λογισμικού δεν είναι πιθανό να οδηγήσουν σε ιδιαίτερα ταχύτερους (ιδίως σε ό,τι αφορά τον κώδικα ακεραίων), μικρότερους (όσον αφορά τον αριθμό των transistors ή το μέγεθος κύβου), πιο απλούς επεξεργαστές ή επεξεργαστές με υψηλότερη αποδοτικότητα ισχύος. Τα τελευταία πέντε έτη έχει καταστεί σαφές ότι η αρχιτεκτονική IA-64 δεν αποτελεί σημαντική εξέλιξη είτε όσον αφορά την αύξηση του ILP είτε αναφορικά με την αντιμετώπιση των προβλημάτων της πολυ-

πλοκότητας και της κατανάλωσης ισχύος των επεξεργασιών υψηλής απόδοσης. Το Παράρτημα Θ διερευνά την εκτίμηση αυτή με περισσότερες λεπτομέρειες.

Οι περιορισμοί της πολυπλοκότητας και της φθίνουσας απόδοσης των επεξεργασιών μεγαλύτερου εύρους έκδοσης πιθανότατα συνεπάγεται, επίσης, ότι είναι εφικτή μόνο η περιορισμένη χρήση της ταυτόχρονης πολυνημάτωσης. Απλά, δεν αξίζει τον κόπο η κατασκευή των επεξεργασιών πολύ μεγάλου εύρους έκδοσης ώστε να δικαιολογεί τις πιο εντατικές υλοποιήσεις της SMT. Για το λόγο αυτό, οι υφιστάμενες σχεδιάσεις έχουν χρησιμοποιήσει μέτριο βεληνεκούς εκδόσεις της SMT με δύο περιβάλλοντα ή απλή πολυνημάτωση δύο περιβαλλόντων, που αποτελούν τις κατάλληλες επιλογές για επεξεργαστές μονής - ή διπλής - έκδοσης.

Οι αρχιτέκτονες, αντί να επιζητούν ολοένα και μεγαλύτερο βαθμό ILP, στρέφουν ολοένα και περισσότερο την προσοχή τους στον TLP των πολυεπεξεργασιών ενός μόνο chip, τον οποίο εξετάζουμε στο επόμενο κεφάλαιο. Το 2000, η IBM ανακοίνωσε τον πρώτο εμπορικό πολυεπεξεργαστή γενικού σκοπού του ενός chip, τον Power 4, ο οποίος περιελάμβανε δύο επεξεργαστές Power 3 και μια ενσωματωμένη κρυφή μνήμη δεύτερου επιπέδου. Από τότε, η Sun Microsystems, η AMD και η Intel έχουν στρέψει το ενδιαφέρον τους στους πολυεπεξεργαστές του ενός μόνο chip και όχι στους πιο εντατικούς μονούς επεξεργαστές.

Το ερώτημα που αφορά την κατάλληλη εξισορρόπηση του ILP και του TLP δεν έχει απαντηθεί με σαφήνεια μέχρι το 2005 και οι σχεδιαστές εξετάζουν μεγάλο εύρος επιλογών, από την απλή διασωλήνωση σε συνδυασμό με τη χρήση περισσότερων επεξεργασιών ανά chip έως και την εντατική εφαρμογή του ILP και της SMT σε συνδυασμό με λιγότερους επεξεργαστές. Επίσης, αυτό που ενδέχεται να αποτελεί την κατάλληλη επιλογή για την αγορά των εξυπηρετητών, οι οποίοι μπορούν να αξιοποιήσουν τον TLP σε μεγαλύτερο βαθμό, μπορεί να διαφέρει από την καλύτερη επιλογή για τους επιτραπέζιους υπολογιστές, στους οποίους η απόδοση του ενός μόνο νήματος μπορεί να εξακολουθεί να αποτελεί πρωταρχική απαίτηση. Θα επιστρέψουμε στο ζήτημα αυτό στο επόμενο κεφάλαιο.

3.9 Ιστορική Προοπτική και Πρόσθετες Πηγές

Η Ενότητα Κ.4 του συνοδευτικού CD περιλαμβάνει την εξέταση της ανάπτυξης της διασωλήνωσης και του παραλληλισμού επιπέδου εντολών. Παρέχουμε πολυάριθμες πηγές για την περαιτέρω εξέταση και διερεύνηση των ζητημάτων αυτών.

Μελέτες Περίπτωσης με Ασκήσεις από τους Wen-mei W. Hwu και John W. Sias

Έννοιες που αναλύονται σε αυτήν τη μελέτη περίπτωσης

- Περιορισμένος ILP εξαιτίας εξαρτήσεων λογισμικού
- Βαθμός ILP που μπορεί να επιτευχθεί έχοντας ως δεδομένους τους περιορισμούς των πόρων του υλικού
- Μεταβλητότητα του ILP εξαιτίας της αλληλεπίδρασης του λογισμικού και του υλικού
- Αλληλοεξαρτώμενη σχέση μεταξύ των τεχνικών του ILP κατά το χρόνο μεταγλώττισης και κατά το χρόνο εκτέλεσης

Μελέτη Περίπτωσης: Εξαρτήσεις και Παραλληλισμός Επιπέδου Εντολών

Στόχος αυτής της μελέτης περίπτωσης είναι η παρουσίαση της αλληλεπίδρασης των παραγόντων που σχετίζονται με το λογισμικό και το υλικό, οι οποίοι οδηγούν στην παράλληλη εκτέλεση επιπέδου εντολών. Αυτή η μελέτη περίπτωσης παρουσιάζει ένα σύντομο παράδειγμα κώδικα, το οποίο απεικονίζει τους διάφορους περιορισμούς του παραλληλισμού επιπέδου εντολών. Ασχολούμενος κανείς με αυτήν τη μελέτη περίπτωσης, θα αποκομίσει γνώσεις για τον τρόπο με τον οποίο αλληλεπιδρούν οι παράγοντες που σχετίζονται με το λογισμικό και το υλικό για τον προσδιορισμό του χρόνου εκτέλεσης ενός συγκεκριμένου τύπου κώδικα κάποιου συστήματος.

Ο πίνακας κατακερματισμού (hash table) αποτελεί ιδιαίτερα δημοφιλή δομή δεδομένων για την οργάνωση μεγάλων συλλογών που περιλαμβάνουν στοιχεία δεδομένων (data items), έτσι ώστε να μπορεί κανείς να απαντά με γρήγορο τρόπο σε ερωτήματα, όπως: «Διαθέτει η συλλογή κάποιο στοιχείο με τιμή ίση με 100;» Αυτό πραγματοποιείται αναθέτοντας τα στοιχεία δεδομένων σε έναν από το μεγάλο αριθμό κάδων (buckets) που υπάρχουν, ανάθεση η οποία υλοποιείται σύμφωνα με την τιμή της συνάρτησης κατακερματισμού (hash function) η οποία παράγεται από τις τιμές των δεδομένων. Τα στοιχεία δεδομένων κάθε κάδου είναι συνήθως οργανωμένα ως συνδεδεμένη λίστα (linked list), ταξινομημένη με κάποια δεδομένη σειρά. Η διαδικασία αναζήτησης στοιχείων εντός του πίνακα κατακερματισμού ξεκινά με τον προσδιορισμό του κάδου που αντιστοιχεί στην τιμή των δεδομένων του ερωτήματος. Στη συνέχεια ακολουθεί η διάσχιση της συνδεδεμένης λίστας των στοιχείων δεδομένων του κάδου και πραγματοποιείται έλεγχος για να εξεταστεί αν υπάρχει κάποιο στοιχείο στη λίστα το οποίο διαθέτει την τιμή του ερωτήματος. Όσο διατηρείται το πλήθος των στοιχείων κάθε κάδου


```

1  typedef struct _Element {
2      int value;
3      struct _Element *next;
4  } Element;
5  Element element[N_ELEMENTS], *bucket[1024];
   /* Ο πίνακας element αρχικοποιείται με τα στοιχεία που πρόκειται να εισαχθούν.
   Οι δείκτες του πίνακα bucket έχουν αρχικά την τιμή NULL*/

6  for (i = 0; i < N_ELEMENTS; i++)
   {
7      Element *ptrCurr, **ptrUpdate;
8      int hash_index;

   /* Εύρεση της θέσης στην οποία πρόκειται να εισαχθεί το νέο στοιχείο */
9      hash_index = element[i].value & 1023;
10     ptrUpdate = &bucket[hash_index];
11     ptrCurr = bucket[hash_index];
   /* Εύρεση της θέσης της αλυσίδας στην οποία θα εισαχθεί το νέο στοιχείο */
12     while (ptrCurr &&
13            ptrCurr->value <= element[i].value)
14     {
15         ptrUpdate = &ptrCurr->next;
16         ptrCurr = ptrCurr->next;
17     }

   /* Ενημέρωση των δεικτών για την εισαγωγή του νέου στοιχείου στην αλυσίδα */
17     element[i].next = *ptrUpdate;
18     *ptrUpdate = &element[i];
   }

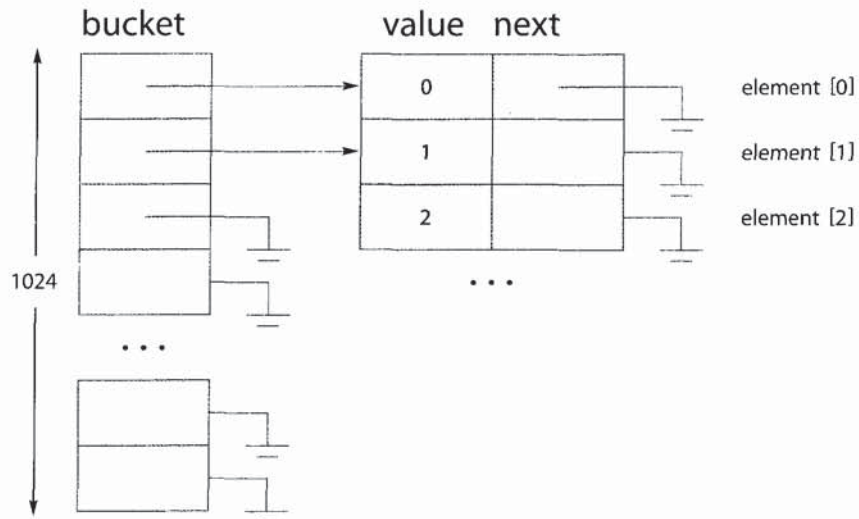
```

Σχήμα 3.14 Παράδειγμα κώδικα πίνακα κατακερματισμού.

σε χαμηλά επίπεδα, το αποτέλεσμα της αναζήτησης μπορεί να προσδιορίζεται πολύ γρήγορα.

Ο κώδικας του Σχήματος 3.14, που είναι γραμμένος σε C, εισάγει μεγάλο αριθμό (N_ELEMENTS) στοιχείων στον πίνακα κατακερματισμού, ο οποίος διαθέτει 1024 κάδους που όλοι τους αποτελούν συνδεδεμένες λίστες ταξινομημένες σε αύξουσα σειρά με βάση την τιμή των στοιχείων. Ο πίνακας element [] περιλαμβάνει τα στοιχεία που πρόκειται να εισαχθούν, τα οποία βρίσκονται στο σωρό. Σε κάθε επανάληψη του εξωτερικού βρόχου (for), που ξεκινά στη γραμμή 6, εισάγεται ένα στοιχείο στον πίνακα κατακερματισμού.

Στη γραμμή 9 του Σχήματος 3.14 υπολογίζεται το hash_index, η τιμή της συνάρτησης κατακερματισμού, με βάση την τιμή των δεδομένων που βρίσκονται



Σχήμα 3.15 Δομή δεδομένων του πίνακα κατακερματισμού.

αποθηκευμένα στο `element [i]`. Η συνάρτηση κατακερματισμού που χρησιμοποιείται είναι πολύ απλή. Αποτελείται από τα 10 λιγότερο σημαντικά bits της τιμής των δεδομένων ενός στοιχείου. Αυτό πραγματοποιείται υπολογίζοντας σε επίπεδο bits το λογικό ΚΑΙ (AND) ανάμεσα στην τιμή των δεδομένων του στοιχείου και τη (δυναδική) μάσκα bit 11 1111 1111 (1023 στο δεκαδικό σύστημα).

Το Σχήμα 3.15 απεικονίζει τη δομή δεδομένων του πίνακα κατακερματισμού που χρησιμοποιείται στο παράδειγμα του κώδικά μας. Ο πίνακας `bucket` που βρίσκεται στο αριστερό τμήμα του Σχήματος 3.15 αποτελεί τον πίνακα κατακερματισμού. Κάθε καταχώριση του πίνακα `bucket` περιλαμβάνει ένα δείκτη προς τη συνδεδεμένη λίστα στην οποία είναι αποθηκευμένα τα στοιχεία δεδομένων του κάδου. Αν ο κάδος `i` είναι κενός την τρέχουσα χρονική στιγμή, η αντίστοιχη καταχώριση `bucket [i]` περιέχει ένα δείκτη NULL. Στο Σχήμα 3.15, καθένας από τους πρώτους τρεις κάδους περιέχει ένα στοιχείο δεδομένων. Οι υπόλοιποι κάδοι είναι κενοί.

Ο βρόχος `while` ξεκινά στη γραμμή 12. Η γραμμή 12 ελέγχει, εξετάζοντας τα περιεχόμενα της μεταβλητής `ptrCurr`, αν υπολείπονται στοιχεία δεδομένων τα οποία πρόκειται να εξεταστούν. Οι γραμμές 13 έως και 16 δεν λαμβάνονται υπόψη σε περίπτωση που δεν υπολείπονται στοιχεία τα οποία πρόκειται να εξεταστούν, είτε διότι ο κάδος είναι κενός είτε επειδή όλα τα στοιχεία δεδομένων της συνδεδεμένης λίστας έχουν εξεταστεί κατά τη διάρκεια των προηγούμενων επαναλήψεων του βρόχου `while`. Στην πρώτη περίπτωση, το νέο στοιχείο δεδομένων εισάγεται στον κάδο ως το πρώτο στοιχείο του. Στη δεύτερη περίπτωση, το νέο στοιχείο εισάγεται ως το τελευταίο στοιχείο της συνδεδεμένης λίστας.

Στην περίπτωση που εξακολουθούν να υπάρχουν στοιχεία προς εξέταση, η γραμμή 13 ελέγχει αν το τρέχον στοιχείο της συνδεδεμένης λίστας περιέχει τιμή η οποία είναι μικρότερη ή ίση με αυτή του στοιχείου δεδομένων που πρόκειται να εισαχθεί στον πίνακα κατακερματισμού. Αν η συνθήκη είναι αληθής, ο βρόχος `while` συνεχίζει να μετακινείται στο επόμενο στοιχείο της συνδεδεμένης λίστας. Οι γραμμές 15 και 16 προχωρούν στο επόμενο στοιχείο της συνδεδεμένης λίστας, μετακινώντας την `ptrCurr` στο επόμενο στοιχείο της συνδεδεμένης λίστας. Διαφορετικά, συνεπάγεται ότι έχει βρεθεί η θέση της συνδεδεμένης λίστας στην οποία πρέπει να εισαχθεί το νέο στοιχείο δεδομένων. Ο βρόχος `while` τερματίζεται και το νέο στοιχείο τοποθετείται ακριβώς πριν το στοιχείο προς το οποίο δείχνει η `ptrCurr`.

Η μεταβλητή `ptrUpdate` προσδιορίζει το δείκτη ο οποίος πρέπει να ενημερωθεί, έτσι ώστε να εισαχθεί το νέο στοιχείο δεδομένων στον κάδο. Ο δείκτης από τη γραμμή 10 έχει οριστεί να παραπέμπει στην καταχώριση του κάδου. Αν ο κάδος είναι κενός, αγνοείται ολόκληρος ο βρόχος `while` και το νέο στοιχείο δεδομένων εισάγεται τροποποιώντας στη γραμμή 18 το δείκτη του `bucket[hash_index]` ώστε αντί της τιμής `NULL` να περιλαμβάνει τη διεύθυνση του νέου στοιχείου δεδομένων. Μετά το βρόχο `while`, η `ptrUpdate` παραπέμπει στο δείκτη ο οποίος πρέπει να ενημερωθεί, έτσι ώστε να εισαχθεί το νέο στοιχείο στον κατάλληλο κάδο.

Αφότου η εκτέλεση ολοκληρώσει το βρόχο `while`, οι γραμμές 17 και 18 ολοκληρώνουν την εργασία εισαγωγής του νέου στοιχείου δεδομένων στη συνδεδεμένη λίστα. Στην περίπτωση που ο κάδος είναι κενός, η `ptrUpdate` παραπέμπει στο `bucket[hash_index]`, το οποίο περιέχει ένα δείκτη `NULL`. Η γραμμή 17 τότε αναθέτει αυτόν το δείκτη `NULL` στον επόμενο δείκτη του νέου στοιχείου δεδομένων. Η γραμμή 18 τροποποιεί το `bucket[hash_table]` ώστε να παραπέμπει στο νέο στοιχείο δεδομένων. Στην περίπτωση που το νέο στοιχείο δεδομένων είναι μικρότερο από όλα τα στοιχεία της συνδεδεμένης λίστας, η `ptrUpdate` παραπέμπει, επίσης, στο `bucket[hash_table]`, το οποίο δείχνει προς το πρώτο στοιχείο της συνδεδεμένης λίστας. Στην περίπτωση αυτή, η γραμμή 17 αναθέτει το δείκτη που παραπέμπει στο πρώτο στοιχείο της συνδεδεμένης λίστας στο νέο δείκτη της νέας δομής δεδομένων.

Στην περίπτωση που το νέο στοιχείο δεδομένων είναι μεγαλύτερο από μερικά και μικρότερο από τα υπόλοιπα στοιχεία της συνδεδεμένης λίστας, η `ptrUpdate` παραπέμπει στον επόμενο δείκτη του στοιχείου μετά το οποίο θα εισαχθεί το νέο στοιχείο δεδομένων. Στην περίπτωση αυτή, στη γραμμή 17 το νέο στοιχείο δεδομένων ενημερώνεται ώστε να παραπέμπει στο στοιχείο που βρίσκεται αμέσως μετά το σημείο εισαγωγής. Στη γραμμή 18 το αρχικό στοιχείο δεδομένων που βρίσκεται ακριβώς πριν το σημείο εισαγωγής ενημερώνεται ώστε να δείχνει προς το νέο στοιχείο δεδομένων. Ο αναγνώστης πρέπει να επαληθεύσει ότι ο κώδικας λειτουργεί με ορθό τρόπο, όταν το νέο στοιχείο δεδομένων πρόκειται να εισαχθεί στο τέλος της συνδεδεμένης λίστας.

Πλέον, αφού έχουμε κατανοήσει τον κώδικα της C, προχωρούμε με την ανάλυση του διαθέσιμου εύρους του παραλληλισμού επιπέδου εντολών που υφίσταται σε αυτό το τμήμα κώδικα.

3.1 [25/15/10/15/20/20/15]<2.1, 2.2, 3.2, 3.3, Παρ. Η> Σε αυτό το τμήμα της μελέτης περίπτωσης θα επικεντρωθούμε στο εύρος του παραλληλισμού επιπέδου εντολών που διαθέτει ο *χρονοδρομολογητής υλικού που ενεργεί κατά το χρόνο εκτέλεσης* εξετάζοντας τα πλέον ιδανικά σενάρια (την ιδανική περίπτωση). (Στη συνέχεια, θα εξετάσουμε λιγότερο ιδανικά σενάρια για το χρονοδρομολογητή υλικού που ενεργεί κατά το χρόνο εκτέλεσης, καθώς και για το παραλληλισμό που διαθέτει ο χρονοδρομολογητής μεταγλώττισης.) Στο ιδανικό σενάριο, θεωρούμε ότι ο πίνακας κατακερματισμού αρχικά είναι κενός. Ας υποθέσουμε ότι υπάρχουν 1024 νέα στοιχεία δεδομένων, των οποίων οι τιμές αριθμούνται σειριακά από το 0 έως και το 1023, έτσι ώστε καθένα από αυτά να οδεύει προς το δικό του κάδο (αυτό καθιστά το πρόβλημα απλά σε υπόθεση ενημέρωσης γνωστών θέσεων του πίνακα). Το Σχήμα 3.15 παρουσιάζει τα περιεχόμενα του πίνακα κατακερματισμού αφότου έχουν εισαχθεί τα τρία πρώτα στοιχεία, σύμφωνα με αυτήν την «ιδανική περίπτωση». Καθώς στην ιδανική αυτή περίπτωση, η *value* (τιμή) του *element [i]* ισούται απλά με *i*, κάθε στοιχείο εισάγεται στο δικό του κάδο.

Για τις ανάγκες αυτής της μελέτης περίπτωσης, υποθέτουμε ότι κάθε γραμμή του κώδικα του Σχήματος 3.14 καταλαμβάνει έναν κύκλο εκτέλεσης (το ύψος εξαρτήσεών της ισούται με 1), ενώ, για τις ανάγκες υπολογισμού του ILP, κάθε γραμμή απαιτεί μία εντολή. Αυτές οι (μη ρεαλιστικές) υποθέσεις γίνονται για την απλοποίηση σε μεγάλο βαθμό της διαδικασίας καταγραφής που απαιτείται για την επίλυση των παρακάτω ασκήσεων. Αξίζει να σημειωθεί ότι οι εντολές *for* και *while* εκτελούνται σε κάθε επανάληψη του αντίστοιχου βρόχου τους, έτσι ώστε να ελέγχεται αν ο βρόχος πρέπει να συνεχίσει. Στην ιδανική αυτή περίπτωση, οι περισσότερες εξαρτήσεις της ακολουθίας του κώδικα έχουν καταστεί λιγότερο αυστηρές και ως εκ τούτου καθίσταται ταχύτατα διαθέσιμος υψηλός βαθμός ILP. Στη συνέχεια, θα εξετάσουμε μια πιο γενική περίπτωση, στην οποία οι ρεαλιστικές εξαρτήσεις της ακολουθίας του κώδικα μειώνουν το εύρος του διαθέσιμου παραλληλισμού.

Επιπλέον, ας υποθέσουμε ότι ο κώδικας εκτελείται σε «ιδανικό επεξεργαστή» που διαθέτει απεριόριστο εύρος έκδοσης, απεριόριστες δυνατότητες μετονομασίας, «πλήρη» γνώση της αποσαφήνισης της προοπείλασης της μνήμης, πρόβλεψη διακλάδωσης κ.ο.κ., έτσι ώστε η εκτέλεση των εντολών να περιορίζεται μόνο εξαιτίας της εξάρτησης δεδομένων. Ας εξετάσουμε τα παρακάτω παραδείγματα εντός αυτού του πλαισίου:

- α. [25]<2.1> Περιγράψτε τις εξαρτήσεις δεδομένων (πραγματικές εξαρτήσεις, αντεξαρτήσεις και εξαρτήσεις εξόδου) και ελέγχου που επηρεάζουν αποφασιστικά τον παραλληλισμό αυτής της ακολουθίας κώδικα, υπό το πρίσμα του χρονοδρομολογητή υλικού που ενεργεί κατά το χρόνο εκτέλεσης. Επισημάνετε μόνο τις *υπαρκτές* εξαρτήσεις (δηλ. αγνοήστε τις εξαρτήσεις ανάμεσα στις εντολές αποθήκευσης και φόρτωσης που προσπελάζουν διαφορετικές διευθύνσεις, ακόμα και αν ο μεταγλωττιστής ή ο επεξεργαστής δεν μπορεί να το προσδιορίσει αυτό ρεαλιστικά). Σχεδιάστε το γράφημα των *δυναμικών* εξαρτήσεων για έξι συνεχόμενες επαναλήψεις του εξωτερικού βρόχου (για την εισαγωγή έξι στοιχείων), για την ιδανική περίπτωση. Προσέξτε ότι σε αυτό το γράφημα των δυναμικών εξαρτήσεων, προσπαθούμε να προσδιορίσουμε τις εξαρτήσεις δεδομένων ανάμεσα σε δυναμικά στιγμιότυπα των εντολών: κάθε στατική εντολή του αρχικού προγράμματος διαθέτει πολλαπλά δυναμικά στιγμιότυπα εξαιτίας της εκτέλεσης του βρόχου. *Συμβουλή*: Οι παρακάτω ορισμοί μπορεί να σας βοηθήσουν στην εύρεση των εξαρτήσεων που σχετίζονται με κάθε εντολή:
- *Πραγματική εξάρτηση δεδομένων*: Από ποιας προηγούμενης εντολής τα αποτελέσματα εξαρτάται άμεσα κάθε εντολή;
 - *Αντεξάρτηση δεδομένων*: Ποιες εντολές εγγράφουν ακολούθως θέσεις που έχουν αναγνωστεί από την εντολή;
 - *Εξάρτηση δεδομένων εξόδου*: Ποιες εντολές εγγράφουν ακολούθως θέσεις που έχουν εγγραφεί από την εντολή;
 - *Εξάρτηση ελέγχου*: Από ποιες προηγούμενες αποφάσεις εξαρτάται η εκτέλεση κάποιας συγκεκριμένης εντολής (σε ποια περίπτωση θα φτάσουμε σε αυτή);
- β. [15]<2.1> Θεωρώντας ότι ισχύει η ιδανική περίπτωση που περιγράφηκε παραπάνω και χρησιμοποιώντας το γράφημα των δυναμικών εξαρτήσεων, απαντήστε στο ερώτημα: Πόσες εντολές εκτελέστηκαν και σε πόσους κύκλους;
- γ. [10]<3.2> Ποιο είναι το μέσο επίπεδο του διαθέσιμου παραλληλισμού κατά τη διάρκεια εκτέλεσης του βρόχου `for`;
- δ. [15]<2.2, Παρ. Η> Στο υποερώτημα (γ) εξετάσαμε το μέγιστο παραλληλισμό που μπορεί να επιτευχθεί από το χρονοδρομολογητή υλικού που ενεργεί κατά το χρόνο εκτέλεσης χρησιμοποιώντας τον κώδικα με τη μορφή που έχει γραφεί. Με ποιον τρόπο μπορεί ο μεταγλωττιστής να αυξήσει το διαθέσιμο παραλληλισμό, με την προϋπόθεση ότι ο μεταγλωττιστής γνωρίζει ότι χειρίζεται την ιδανική περίπτωση. *Συμβουλή*: Αναλογιστείτε ποιος είναι ο κύριος περιορισμός που παρεμποδίζει την ταυτόχρονη εκτέλεση περισσότερων επαναλήψεων στην

ιδανική περίπτωση. Με ποιον τρόπο μπορεί να δομηθεί εκ νέου ο βρόχος, προκειμένου ο περιορισμός αυτός να υφίσταται σε μικρότερο βαθμό;

- ε. [25]<3.2, 3.3> Για λόγους απλότητας, θεωρήστε ότι μόνο οι μεταβλητές `i`, `hash_index`, `ptrCurr` και `ptrUpdate` χρειάζεται να καταλαμβάνουν καταχωρητές. Υποθέτοντας ότι ισχύει γενική μετονομασία, ποιος είναι ο αριθμός των απαραίτητων καταχωρητών για την επίτευξη του μέγιστου δυνατού παραλληλισμού του υποερωτήματος (β);
- στ. [25]<3.3> Υποθέστε ότι στην απάντησή σας στο υποερώτημα (α) υπάρχουν 7 εντολές σε κάθε επανάληψη. Με βάση το παραπάνω, καθώς και την υπόθεση ότι η χρονοδρομολόγηση των εντολών πραγματοποιείται με συνέπεια και σταθερότητα και ότι υφίσταται ρυθμός έκδοσης της τάξης των 3 εντολών ανά κύκλο, εξετάστε με ποιον τρόπο επηρεάζεται ο χρόνος εκτέλεσης.
- ζ. [15]<3.3> Τέλος, υπολογίστε το ελάχιστο μέγεθος παραθύρου εντολών που χρειάζεται για την επίτευξη του μέγιστου επιπέδου του παραλληλισμού.

3.2 [15/15/15/10/10/15/15/10/10/10/25]<2.1, 3.2, 3.3> Ας εξετάσουμε, πλέον, τα λιγότερο ιδανικά σενάρια για την επίτευξη παραλληλισμού επιπέδου εντολών με τη βοήθεια του χρονοδρομολογητή υλικού που ενεργεί κατά το χρόνο εκτέλεσης στον κώδικα του πίνακα κατακερματισμού του Σχήματος 3.14 (στη γενική περίπτωση). Ας υποθέσουμε ότι πλέον δεν διασφαλίζεται ότι κάθε κάδος θα λαμβάνει ακριβώς ένα στοιχείο. Ας αξιολογήσουμε εκ νέου την εκτίμησή μας όσον αφορά το διαθέσιμο παραλληλισμό, με δεδομένη την πιο ρεαλιστική κατάσταση που ισχύει, η οποία επιφέρει μερικές επιπρόσθετες σημαντικές εξαρτήσεις.

Ας θυμηθούμε ότι, στην ιδανική περίπτωση, ο σχετικά σειριακός εσωτερικός βρόχος δεν χρησιμοποιούνταν και ότι ο εξωτερικός βρόχος παρείχε ευρύτατο παραλληλισμό. Γενικά, ο εσωτερικός βρόχος χρησιμοποιείται: ο εσωτερικός βρόχος `while` μπορεί να επαναλαμβάνεται μία ή περισσότερες φορές. Ας έχουμε κατά νου ότι ο εσωτερικός βρόχος, δηλαδή ο βρόχος `while`, διαθέτει περιορισμένο μόνο εύρος παραλληλισμού επιπέδου εντολών. Πρώτον, κάθε επανάληψη του βρόχου `while` εξαρτάται από το αποτέλεσμα της προηγούμενης επανάληψης. Δεύτερον, εντός κάθε επανάληψης, εκτελείται μόνο ένας μικρός αριθμός εντολών.

Αντιθέτως, ο εξωτερικός βρόχος είναι αρκετά παράλληλος. Η εισαγωγή δύο στοιχείων του εξωτερικού βρόχου μπορεί να πραγματοποιείται παράλληλα, όσο τα στοιχεία αυτά τοποθετούνται σε διαφορετικούς κάδους. Ακόμα και στην περίπτωση που τα στοιχεία αυτά τοποθετούνται στον ίδιο κάδο,

η εισαγωγή τους μπορεί να εκτελείται παράλληλα με την προϋπόθεση ότι υφίσταται κάποιο είδος αποσαφήνισης της μνήμης που επιβάλλει την ορθότητα των εντολών φόρτωσης και αποθήκευσης, οι οποίες εκτελούνται εκ μέρους κάθε στοιχείου.

Αξίζει να σημειωθεί ότι, στην πραγματικότητα, οι τιμές των στοιχείων δεδομένων είναι πιθανό να είναι κατανομημένες τυχαία. Παρόλο που έχουμε ως στόχο να παράσχουμε στον αναγνώστη πληροφορίες σχετικά με τα πιο ρεαλιστικά σενάρια εκτέλεσης, θα ξεκινήσουμε με συνηθισμένα, αλλά όχι ιδανικά, πρότυπα τιμών δεδομένων, τα οποία επιδέχονται συστηματική ανάλυση. Αυτά τα πρότυπα τιμών παρέχουν κάποια ενδιάμεσα βήματα που διευκολύνουν την κατανόηση του εύρους του παραλληλισμού επιπέδου εντολών, καθώς μετακινούμαστε στις πιο γενικές, τυχαίες τιμές δεδομένων.

- α. [15]<2.1> Σχεδιάστε το γράφημα των δυναμικών εξαρτήσεων για τον κώδικα του πίνακα κατακερματισμού του Σχήματος 3.14, όταν οι τιμές των 1024 στοιχείων που πρόκειται να εισαχθούν είναι οι 0, 1, 1024, 1025, 2048, 2049, 3072, 3073, Περιγράψτε τις νέες εξαρτήσεις που υπάρχουν μεταξύ των επαναλήψεων του βρόχου `for`, όταν ο βρόχος `while` επαναλαμβάνεται μία ή περισσότερες φορές. Δώστε ιδιαίτερη προσοχή στο γεγονός ότι ο εσωτερικός βρόχος `while` εκτελείται πλέον μία ή περισσότερες φορές. Ο αριθμός των εντολών του εξωτερικού βρόχου `for` είναι ως εκ τούτου πιθανό να διαφέρει μεταξύ των διάφορων επαναλήψεων. Για τις ανάγκες του προσδιορισμού των εξαρτήσεων μεταξύ των εντολών φόρτωσης και αποθήκευσης, θεωρήστε ότι εφαρμόζεται δυναμική αποσαφήνιση της μνήμης, η οποία δεν είναι σε θέση να επιλύσει τις εξαρτήσεις μεταξύ δύο προσπελάσεων της μνήμης σε διαφορετικούς καταχωρητές δείκτη βάσης. Για παράδειγμα, το υλικό που ενεργεί κατά το χρόνο εκτέλεσης δεν μπορεί να προβεί στην αποσαφήνιση μιας εντολής αποθήκευσης που βασίζεται στην `ptrUpdate` και μίας εντολής φόρτωσης που βασίζεται στην `ptrCurr`.
- β. [15]<2.1> Χρησιμοποιώντας το γράφημα των δυναμικών εξαρτήσεων του υποερωτήματος (α), υπολογίστε τον αριθμό των εντολών που θα εκτελεστούν.
- γ. [15]<2.1> Χρησιμοποιώντας το γράφημα των δυναμικών εξαρτήσεων του υποερωτήματος (α) και υποθέτοντας ότι υφίσταται απεριόριστος αριθμός πόρων υλικού, υπολογίστε τον αριθμό των κύκλων που απαιτούνται για την εκτέλεση όλων των εντολών που υπολογίσατε στο υποερώτημα (β).
- δ. [10]<2.1> Ποιος είναι ο βαθμός του διαθέσιμου παραλληλισμού επιπέδου εντολών του γραφήματος δυναμικών εξαρτήσεων του υποε-

ρωτήματος (α);

- ε. [10]<2.1, 3.2> Κάνοντας την ίδια υπόθεση σε ό,τι αφορά το μηχανισμό αποσαφήνισης της μνήμης κατά το χρόνο εκτέλεσης του υποερωτήματος (α), επισημάνετε κάποια ακολουθία στοιχείων δεδομένων η οποία θα επιφέρει την ισχύ του σεναρίου χειρότερης περίπτωσης όσον αφορά τον τρόπο με τον οποίο οι νέες αυτές εξαρτήσεις επηρεάζουν το διαθέσιμο επίπεδο του παραλληλισμού.
- στ. [15]<2.1, 3.2> Πλέον, χρησιμοποιήστε την ακολουθία του σεναρίου χειρότερης περίπτωσης του υποερωτήματος (ε), εξηγήστε την επίδραση που ενδέχεται να έχει η χρησιμοποίηση ενός απόλυτα ακριβούς μηχανισμού αποσαφήνισης (δηλ. ενός συστήματος το οποίο ανιχνεύει όλες τις εκκρεμείς εντολές αποθήκευσης και επιτρέπει σε όλες τις εντολές φόρτωσης που δεν οδηγούν σε σύγκρουση να συνεχίζουν). Υπολογίστε τον αριθμό των κύκλων ρολογιού που απαιτούνται για την εκτέλεση των εντολών του γραφήματος δυναμικών εξαρτήσεων. Με βάση τα όσα έχετε μάθει έως τώρα, εξετάστε το παρακάτω ζεύγος ποιοτικών ερωτημάτων: Ποια είναι η επίδραση που έχει η παροχή της δυνατότητας της υποθετικής έκδοσης των εντολών φόρτωσης, πριν να είναι γνωστές οι διευθύνσεις των προγενέστερων εντολών αποθήκευσης; Με ποιον τρόπο επηρεάζει αυτού του είδους η εικασία τη βαρύτητα του λανθάνοντος χρόνου της μνήμης για τον κώδικα αυτό;
- ζ. [15]<2.1, 3.2> Προχωρήστε κάνοντας τις ίδιες υποθέσεις με αυτές του υποερωτήματος (στ) και υπολογίστε τον αριθμό των εντολών που εκτελούνται.
- η. [10]<2.1, 3.2> Προχωρήστε κάνοντας τις ίδιες υποθέσεις με αυτές του υποερωτήματος (στ) και υπολογίστε το εύρος του παραλληλισμού επιπέδου εντολών που είναι διαθέσιμος για το υλικό που ενεργεί κατά το χρόνο εκτέλεσης.
- θ. [10]<2.1, 3.2> Στο υποερώτημα (η), ποια είναι η επίδραση που έχουν τα παράθυρα εντολών με περιορισμένο μέγεθος στο επίπεδο του παραλληλισμού επιπέδου εντολών;
- ι. [10]<3.2, 3.3> Πλέον, συνεχίζοντας την εξέταση της επίλυσης του υποερωτήματος (η), περιγράψτε τα αίτια των αστοχιών της πρόβλεψης διακλάδωσης και την επίδραση που έχει κάθε πρόβλεψη διακλάδωσης στο επίπεδο του διαθέσιμου παραλληλισμού. Περιγράψτε εν συντομία τις επιπτώσεις που υφίστανται αναφορικά με την ισχύ και την αποδοτικότητα. Ποιο είναι το ενδεχόμενο κόστος και ποια τα οφέλη που μπορεί να προκύψουν από την εκτέλεση πολλών υποθετικών εντολών εκτός διαδρομής (δηλ. από την έναρξη της εκτέλεσης εντολών οι οποίες θα απορριφθούν μέσω της ανίχνευσης της εσφαλμένης πρόβλεψης διακλαδώσεων); *Συμβουλή*: Αναλογιστείτε τις επιπτώσεις που

μπορεί να έχει για την εκτέλεση επακόλουθων λειτουργιών εισαγωγής, η εσφαλμένη πρόβλεψη του αριθμού των στοιχείων που πρέπει να τοποθετηθούν πριν το σημείο εισαγωγής.

- κ. [25]<3> Εξετάστε την ιδέα χρήσης γραφήματος στατικών εξαρτήσεων, το οποίο απεικονίζει όλες τις εξαρτήσεις του σεναρίου χειρότερης περίπτωσης, έχοντας ως απώτερο στόχο τον περιορισμό της χρονοδρομολόγησης και της βελτιστοποίησης του μεταγλωττιστή. Σχεδιάστε το γράφημα των στατικών εξαρτήσεων για τον πίνακα κατακερματισμού που απεικονίζεται στο Σχήμα 3.14.

Συγκρίνετε το γράφημα των στατικών εξαρτήσεων με τα διάφορα γραφήματα δυναμικών εξαρτήσεων που σχεδιάσατε προηγουμένως. Περιγράψτε σε μία ή δύο παραγράφους τα αποτελέσματα της σύγκρισης αυτής για τη δυναμική και τη στατική ανακάλυψη του παραλληλισμού επιπέδου εντολών, χρησιμοποιώντας τον κώδικα του πίνακα κατακερματισμού του παραδείγματος αυτού. Συγκεκριμένα, με ποιον τρόπο περιορίζεται ο μεταγλωττιστής λαμβάνοντας υπόψη συνεχώς τη χειρότερη περίπτωση, όπου ο μηχανισμός υλικού ενδέχεται να είναι ελεύθερος να αξιοποιεί ευκαιριακά τυχαίες περιπτώσεις; Ποιες προσεγγίσεις μπορεί να βοηθήσουν το μεταγλωττιστή, ώστε να προβαίνει σε καλύτερη χρήση του κώδικα αυτού;