Παραλληλοποίηση της αναζήτησης μέγιστου και ελάχιστου στοιχείου σε μονοδιάστατο πίνακα

Νίκος Λιθαρής Π2019083

Contents

Παραλληλοποίηση με pthreads

Ο κώδικας χρησιμοποιεί τη βιβλιοθήκη pthreads για να πετύχει την παραλληλοποίηση των εργασιών. Αρχικοποιεί έναν πίνακα τυχαίων αριθμών, τον χωρίζει σε πολλαπλά τμήματα και αναθέτει κάθε τμήμα σε ένα thread. Κάθε thread αναζητά το μέγιστο στοιχείο εντός του τμήματος που του έχει ανατεθεί. Στη συνέχεια, το thread ενημερώνει `maxval` χρησιμοποιώντας ένα mutex για να εξασφαλίσει την ασφάλεια του thread. Τέλος, η μέγιστη τιμή που βρίσκει κάθε thread συγκρίνεται, και προσδιορίζεται το συνολικό μέγιστο.

Αποτελέσματα

minmax	minmax-pthreads
Exec Time (sec) = 0.052406	Exec Time (sec) = 0.072923

Ο Αλγόριθμος `minmax` είναι κατα μέσο όρο 39.15% γρηγορότερος σε σχέση με τον `minmax-pthreads`

Παραλληλοποίηση με OpenMP

Ο κώδικας χρησιμοποιεί τις εντολες της βιβλιοθήκης OpenMP για τον παραλληλισμό του υπολογισμού. Ο κώδικας αρχικοποιεί έναν πίνακα τυχαίων αριθμών και αναθέτει στο αρχικό στοιχείο τόσο την ελάχιστη (`checkmin`) όσο και τη μέγιστη (`checkmax`) τιμή. Η επανάληψη που γεμίζει τον πίνακα παραλληλοποιείται χρησιμοποιώντας την εντολή `#pragma omp parallel for`. Οι εντολές `reduction(min: minval)` και `reduction(max: maxval)` χρησιμοποιούνται στην επόμενη επανάληψη για την εκτέλεση παράλληλων πράξεων εύρεσης των τελικών ελάχιστων και μέγιστων τιμών. Τέλος, οι υπολογισμένες ελάχιστες και μέγιστες τιμές συγκρίνονται με τις αναμενόμενες τιμές (`checkmin` και `checkmax`) για την επαλήθευση της ορθότητας του υπολογισμού.

Αποτελέσματα

minmax	minmax-openmp
Exec Time (sec) = 0.052406	Exec Time (sec) = 0.018248

Ο Αλγόριθμος `minmax` είναι κατα μέσο όρο 65.18% πιο αργός σε σχέση με τον `minmax-openmp`

Παραλληλοποίηση με Intel TBB

Ο κώδικας είναι μια παράλληλη υλοποίηση για την εύρεση του ελάχιστου και του μέγιστου στοιχείου σε ένα μονοδιάστατο διάνυσμα `v` με τη χρήση της βιβλιοθήκης Intel Threading Building Blocks (TBB). Χρησιμοποιεί τον βρόχο `parallel_for` του TBB για να χωρίσει το διάνυσμα σε μπλοκ και να αναθέσει κάθε μπλοκ σε ξεχωριστό thread. Κάθε thread διατηρεί μια τοπική ελάχιστη και μέγιστη τιμή χρησιμοποιώντας τα `tbb::combinable` και `minmax_combinable` αντίστοιχα. Η συνάρτηση lambda αρχικοποιεί τις τοπικές ελάχιστες και μέγιστες τιμές με το πρώτο στοιχείο του διανύσματος. Καθώς κάθε thread επεξεργάζεται το μπλοκ που του έχει ανατεθεί, ενημερώνει τις τοπικές ελάχιστες και μέγιστες τιμές. Μόλις ολοκληρωθεί η επανάληψη, οι τοπικές ελάχιστες και μέγιστες τιμές από κάθε thread συνδυάζονται χρησιμοποιώντας τη μέθοδο `combine()` της `minmax_combinable` για να εκτελεστεί η πράξη μείωσης. Τέλος, τα αποτελέσματα του παράλληλου υπολογισμού συγκρίνονται με τον σειριακό υπολογισμό για να διασφαλιστεί η ορθότητα.

Αποτελέσματα

minmax	minmax-tbb
Exec Time (sec) = 0.052406	Exec Time (sec) = 0.02112

Ο Αλγόριθμος `minmax` είναι κατα μέσο όρο 59.7% πιο αργός σε σχέση με τον `minmax-tbb`