UNIVERSITÀ DEGLI STUDI FIRENZE

Architetture Avanzate

Rankboost: implementazione e testing di un algorimo learning-to-rank

Tommaso Papini
tommaso.papini1@stud.unifi.it

Gabriele Bani
gabriele.bani@stud.unifi.it

IMPLEMENTAZIONE

Il progetto che abbiamo realizzato prevede l'implementazione dell'algoritmo RankBoost all'interno del framework quickrank, utilizzando C++ come linguaggio di programmazione.
Dopo aver realizzato una versione sequenziale dell'algoritmo, abbiamo provveduto a parallelizzare alcune porzioni del codice utilizzando il tool OpenMp. In alcuni casi, infatti, le versioni sequenziali di alcune porzioni del codice risultano migliori in termini di velocità di esecuzione rispetto alle rispettive versioni parallele e, pertanto, non è stato necessario parallelizzarle.
Abbiamo provveduto anche ad analizzare le prestazioni dell'algoritmo utilizzando le varie tipologie di scheduling parallelo: runtime, guided, dynamic e static. Lo scheduling guided risulta il migliore tra le varie tipologie di scheduling e, per questo motivo, abbiamo provveduto a modificare la variabile di sistema OMP_SCHEDULE in modo che tutte le porzioni di codice parallelizzate utilizzino lo scheduling di tipo guided. Similmente, abbiamo provveduto ad utilizzare la variabile n_thread (dichiarata in custom_ltr.h) per poter gestire più semplicemente il numero di thread creati in ogni porzione di codice parallela.

PROBLEMI INCONTRATI

I principali problemi riscontrati nell'implementare l'algoritmo RankBoost, all'interno del framework quickrank, sono sorti all'interno del metodo compute_weak_ranker. Parallelizzare il ciclo più esterno, ad esempio, portava a risultati differenti (e spesso erronei) rispetto alla versione sequenziale dell'algoritmo. Ciò non avveniva, tuttavia, se il ciclo parallelizzato era il ciclo più interno. In tal caso, infatti,

sono stati ottenuti gli stessi risultati della versione sequenziale, pur osservando uno speedup notevole. Tale problematica è stata risolta effettuando una gestione più opportuna delle variabili condivise. Una mal gestione delle variabili condivise aveva portato anche ad ottenere risultati non deterministici sia in termini di tempo di esecuzione dell'algoritmo, sia in termini di score. Non dichiarando la Feature "Feat" di tipo firstprivate all'interno della dichiarazione della direttiva pragma omp, infatti, ottenevamo tali risultati.

PARALLELIZZAZIONE

Si toglie???

SPEEDUP E MIGLIORAMENTI

TESTING