

Instituto Superior Técnico – Universidade de Lisboa

Ano Lectivo 2014 / 2015

1º Semestre

Computação Paralela e Distribuída

**Computação Paralela e Distribuída**

Projecto

**Longest Common Subsequence**

Grupo nº26

**Daniel Teles Arrais,** nº 69675

**Miguel Nobre da Costa**, nº73359

**Ricardo Filipe Amendoeira**, nº7337

Método de paralelização

A nossa implementação série usa um algoritmo de computação da matriz (antes de encontrar a sub-sequência) que pelas nossas análises se mostrou ser de longe a componente mais computacionalmente exigente do programa e por essa razão foi nessa parte que nos focámos em paralelizar.

A computação da matriz tem complexidade O(n\*m) em que n e m são os comprimentos das duas strings e cada célula da matriz depende do valor das células imediatamente acima e à esquerda e da célula diagonal anterior, o que dificulta a paralelização. A solução escolhida de entre as vàrias que experimentàmos foi a de resolver a matriz por antidiagonais, o que nos permite calcular as células internas a cada antidiagonal por uma ordem arbitrária e cumprir as dependências.

Implementação

Usámos OpenMP para paralelizar o *for* interior (percorrer células da antidiagonal respectiva) e deixámos o *for* exterior (percorrer antidiagonais da matriz) em execução série.

Dentro deste método impletámos também alguns melhoramentos como [melhoramentos, como inverter a ordem das threads, se ajudar].

Assim conseguimos um speedup S = Ts/Tp de [speedup] usando os quatro cores do computador do laboratório.

Resultados

Este resultado está dentro do esperado uma vez que atingir um speedup

S = Ts/Ncores é impossível devido ao overhead de criar e terminar threads e distribuir o trabalho pelas threads e os acessos à memória também criam overheads que limitam a performance do paralelismo.