Circuit Satisfiability Problem – Inleveropgave
Analyse van de resultaten

Sinem Ertem

In deel I moeten we een baseline opzetten waarin we het *circuitSatisfiabilityProblem.c* bestand runnen met 1 proces. Vervolgens moeten we de verstreken tijd meten met een MPI-functie.

Tabel 1

Number of processes	***Time in seconds***	***Solutions found***
1	*259.30*	*81*

In deel II wordt hetzelfde bestand verder uitgewerkt met een parallel loop patroon uit het *parallelLoopSlices.c* bestand. Door het gebruik van meer dan 1 proces kunnen we namelijk (als het goed is) de tijd die het programma nodig heeft om volledig te runnen, verkorten.

Het eerste bestand moet aan de hand van berekeningen 2^{32} mogelijkheden uitvoeren. In mijn baseline met 1 proces, kost het bijna 260 seconden om 81 oplossingen te vinden. Dit breiden we uit tot [2,4,8,16] processen.

Tabel 2

Number of processes	***Time in seconds***	***Solutions found***
2	*159.98*	*81*
4	*144.60*	*81*
8	*129.30*	*81*
16	*132.68*	*81*

Zoals je kunt aflezen in tabel 2 neemt de tijd in seconden af zodra je meer processen gebruikt. Door het gebruiken van 8 processen in vergelijking met de baseline is het zelfs **gehalveerd.** Zodra we gebruik maken van 16 processen zie je dat de tijd vervolgens weer begint toe te nemen en kunnen we daarmee concluderen dat het gebruik van 8 processes het efficiënts is. Het is mij niet gelukt om het programma met [32,64,128,256] processen te runnen en daarmee kan ik mijn conclusie niet volledig onderbouwen.

Grafiek 1: line-chart verzamelde data