Inleveropgave 4: Circuit Satisfiability Problem

Leerling: Sietse Neve

Leerlingnummer: 1810364

Github link: https://github.com/DrZisnotavailable/HPP.git

Doel van het experiment

Het doel van deze opdracht was om het brute-force algoritme voor het oplossen van het circuit satisfiability probleem te versnellen door middel van parallelisatie met MPI. Hierbij is zowel een initiële versie (Deel 1) als een geoptimaliseerde versie gebaseerd op het parallel loop patroon (Deel 2) uitgevoerd. Voor beide versies is de tijd gemeten bij verschillende aantallen processen.

Meetgegevens

Deel 1: Initieel parallel programma

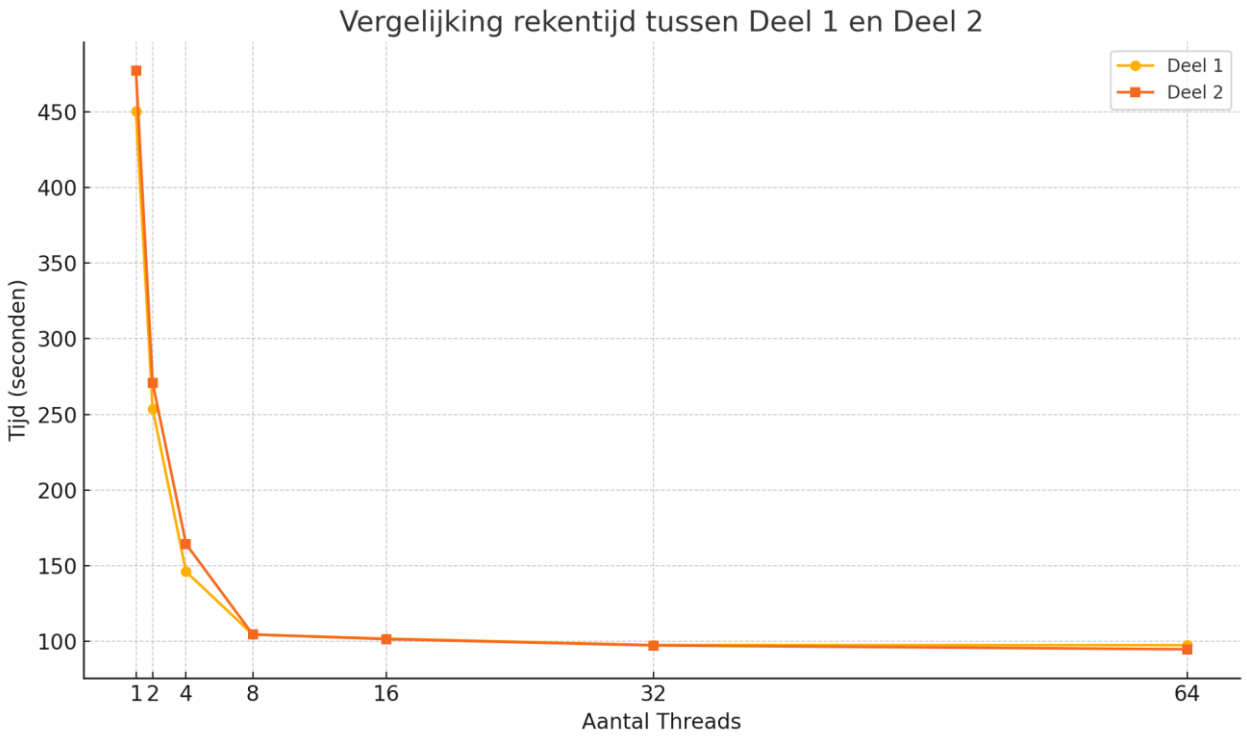
Number of Processes	Time (sec)	Solutions
1	450.271	81
2	253.727	81
4	146.028	81
8	104.488	81
16	101.790	81
32	97.5393	81
64	97.4458	81

Deel 2: Met parallel loop patroon (chunks)

Number of Processes	Time (sec)	Solutions
1	477.186	81
2	270.889	81
4	164.356	81
8	104.509	81
16	101.472	81
32	97.2788	81
64	94.7388	81

Grafische weergave

Onderstaande grafiek toont het verloop van de uitvoeringstijd (in seconden) ten opzichte van het aantal processen voor zowel Deel 1 als Deel 2:



De grafiek maakt duidelijk dat de tijdswinst significant is tot en met ongeveer 8 processen, daarna vlakt de verbetering af. De daling is aanvankelijk steil, wat duidt op effectieve parallelisatie. De afvlakking wijst op overhead en beperkte schaalbaarheid bij deze brute-force aanpak.

Analyse en conclusie

Beide implementaties tonen een significante versnelling bij het gebruik van meerdere processen. De baseline (1 process) kost meer dan 7 minuten (450-477 seconden). Vanaf 8 processen is de snelheidstoename merkbaar minder; de grafiek vlakt hier duidelijk af.

Het gebruik van `MPI_Reduce` werkt correct: bij elk aantal processen blijft het gevonden aantal oplossingen 81, wat gelijk is aan het verwachte aantal. Er is dus geen sprake van race conditions of foutieve werkverdeling.

Het verschil tussen Deel 1 en Deel 2 is marginaal: Deel 2 is niet structureel sneller, ondanks de verbeterde werkverdeling via chunking. Dit suggereert dat de overhead van het opstarten van MPI-processen en communicatie groter is dan de winst van de verbeterde balans in workload.

Conclusie: Parallelisatie via MPI levert duidelijke tijdswinst op, met name bij het opschalen van 1 naar 8 processen. Meer processen daarna leveren minder voordeel op. De chunking-

benadering is theoretisch beter, maar levert praktisch weinig extra winst bij deze specifieke brute-force workload. Beide implementaties zijn correct en schaalbaar, met beperkte maar nuttige snelheidswinst.