Joey Welvaadt

Hogeschool utrecht  1734098

https://github.com/JoeyWelvaadt1999/HPPv2

4. Circuit satisfiability

# Code

A screen shot of a computer program

AI-generated content may be incorrect.

# Results

Er worden in dit hoofdstuk 2 verschillende grafieken weergegeven: de tijd die het met elk aantal processen heeft gekost en wat daar de speed up factor voor was (van bijvoorbeeld 1 naar 2 processen).

A graph with a line

AI-generated content may be incorrect.

Figuur Processen vs time

A graph with a line and numbers

AI-generated content may be incorrect.

Figuur 2 Speedup factor vs processen

Wat mij is opgevallen bij het verzamelen van deze data is: tot ongeveer 32 processen wordt de tijd dat het kost voor het programma geleidelijk verlaagt, maar van 32 naar 64 processen stijgt het weer een beetje. Het werk wat elk proces moet uitvoeren wordt steeds kleiner, de *relatieve overhead* stijgt. Het komt er op neer dat er een punt is dat het aanmaken van processen en het opdelen van het werk meer tijd kost dan het uiteindelijk uitvoeren van het programma. Zo is ook al te zien bij de speed up factor, dat van 16 naar 32 die 1.5 is. En van 32 naar 64 1.1, vervolgens van 64 naar 128 gaat deze onder 1 (zie tabel 1).

|  |  |  |
| --- | --- | --- |
| **Aantal Processen** | **Tijd (s)** | **Speedup Factor** |
| 1 | 201.835 | 0.00 |
| 2 | 98.2361 | 2.05456 |
| 4 | 52.0982 | 1.88564 |
| 8 | 34.4968 | 1.51024 |
| 16 | 22.0829 | 1.56221 |
| 32 | 19.7724 | 1.11681 |
| 64 | 20.5361 | 0.96281 |
| 128 | 18.489 | 1.11066 |
| 256 | 18.8386 | 0.98144 |

Tabel 1 Uitvoertijd en speedup factor bij verschillende aantallen processen