Protokoll zum Praktikum Äquivalenzprüfung von Schaltkreisen

Fakultät Informatik
TU Dresden

Christian Kroh
Matrikelnummer: 3755154
Studiengang: Informatik (Diplom)
Jahrgang: 2010/2011

25. Januar 2014, Dresden

Inhaltsverzeichnis

0.1 Aufgabe 1 - Code-Optimierung

Listing 1: matmul0.c-Original

```
47        /* Begin matrix matrix multiply kernel */
48        for ( uint32_t i = 0; i < dim; i++ )
49        {
50            for ( uint32_t j = 0; j < dim; j++ )
51            {
52                for ( uint32_t k = 0; k < dim; k++ )
53                {
54                    // C[i][j] += A[i][k] * B[k][j]
55                    C[ i * dim + j ] += A[ i * dim + k ] * B[ k * dim + j ];
56                }
57            }
58        }
```

0.1.1 Optimierungen

1. Speichern öfters genutzter, zusammengesetzter Werte
   `i_mult_dim = i * dim;`
   `i_mult_dim_add_k = i_mult_dim + k;`
   `k_mult_dim = k * dim;`

2. halten des Wertes einer Matrix für Multiplikation
   `A[ i_mult_dim_add_k ]`

Listing 2: matmul0.c-optimiert

```
46    uint32_t i_mult_dim, i_mult_dim_add_k, k_mult_dim, i, j, k;
47
48        /* Begin matrix matrix multiply kernel */
49        for ( i = 0; i < dim; i++ )
50        {
51      i_mult_dim = i * dim;
52            for ( k = 0; k < dim; k++ )
53            {
54        i_mult_dim_add_k = i_mult_dim + k;
55        k_mult_dim = k * dim;
56            for ( j = 0; j < dim; j++ )
57            {
58                    // C[i][j] += A[i][k] * B[k][j]
59                    C[ i_mult_dim + j ] += A[ i_mult_dim_add_k ] * B[ k_mult_dim + j ];
60                }
61            }
62        }
63        /* End matrix matrix multiply kernel */
```

0.1.2 Implementierung

Der verwendete SAT-Solver ist miniSAT in der Version 1.14 als Binary. Die generierten KNF-Klauseln werden in eine Datei mit dem Namen cnf geschrieben und anschliesend vom SAT-Solver gelesen. Die jeweiligen Gatter-Klassen sind so implementiert, dass sie ihre Funktion als KNF darstellen können. Die Klasse `Solver` enthält einen Zähler für die Benennung der KNF-Variablen.

0.2 Aufgabe 2 - Zeitmessungen

(nicht Taurus)

0.2.1 Original

Dimension	Runtime	GFLOP/s
32	0.0002s	0.28
64	0.0019s	0.27
96	0.0067s	0.26
128	0.0181s	0.23
160	0.0348s	0.24
192	0.0603s	0.23
224	0.1030s	0.22
256	0.1808s	0.19
320	0.2984s	0.22
384	0.6227s	0.18
448	0.9420s	0.19
512	1.5226s	0.18
640	3.2836s	0.16
768	6.0007s	0.15
896	10.2551s	0.14
1024	22.3022s	0.10

0.2.2 Speichern öfters genutzter, zusammengesetzter Werte

Dimension	Runtime	GFLOP/s
32	0.0002s	0.27
64	0.0017s	0.30
96	0.0059s	0.30
128	0.0153s	0.27
160	0.0277s	0.30
192	0.0502s	0.28
224	0.0806s	0.28
256	0.1397s	0.24
320	0.2506s	0.26
384	0.4941s	0.23
448	0.7273s	0.25
512	1.1767s	0.23
640	2.6372s	0.20
768	4.7501s	0.19
896	8.2936s	0.17
1024	25.0923s	0.09

0.2.3 halten des Wertes einer Matrix für Multiplikation

Dimension	Runtime	GFLOP/s
32	0.0002s	0.30
64	0.0018s	0.29
96	0.0060s	0.30
128	0.0130s	0.32
160	0.0253s	0.32
192	0.0449s	0.32
224	0.0691s	0.33
256	0.1020s	0.33
320	0.1995s	0.33
384	0.3488s	0.32
448	0.5569s	0.32
512	0.8348s	0.32
640	1.6256s	0.32
768	2.8065s	0.32
896	4.5334s	0.32
1024	6.7665s	0.32