

# Übungen zur Vorlesung “Architektur und Programmierung von Grafik- und Koprozessoren”

## Übungsblatt 7

Sommersemester 2019

### 7 Wiederholung (alte Klausuraufgaben)

#### Aufgabe 7.1

Wir betrachten den nachfolgenden Ausschnitt des Befehlssatzes einer hypothetischen CPU:

Instruktion	Beschreibung	Latenz
ADD \$R1 <i>C</i>	Addiere den konstanten Wert <i>C</i> auf den Inhalt von Register \$R1	1 TZ
ADD \$R1 \$R2	Addiere Registerinhalte \$R1 und \$R2, speichere das Ergebnis in \$R1	1 TZ
MUL \$R1 \$R2	Multipliziere Registerinhalte \$R1 und \$R2, speichere das Ergebnis in \$R1	1 TZ
LD \$R1 [ <i>S</i> ]	Lade den Inhalt an Speicherstelle [ <i>S</i> ] in Register \$R1	1 TZ
ST [ <i>S</i> ] \$R1	Speichere den Inhalt von Register \$R1 an Speicherstelle [ <i>S</i> ]	1 TZ
JNZ \$R1 label1	Falls Inhalt von Register \$R1 $\neq 0$ , springe zu Label label1	1 TZ

sowie das nachfolgende Programm:

---

```
LD $R1 [0xFF-0]
LOOP:
ADD $R1 -1
LD $R2 [0xC0-$R1]
LD $R3 [0x80-$R1]
ADD $R2 $R3
ADD $R2 3
MUL $R2 $R2
ST [0xC0-$R1] $R2
JNZ $R1 LOOP
```

---

Nehmen Sie an, dass der Inhalt der Register \$R1, \$R2 und \$R3 zunächst 0 beträgt und dass im Speicher an der Adresse 0xFF der konstante Wert 2 steht.

a.)

Zeichnen Sie den Datenabhängigkeitsgraph für die Schleife im Programm auf.

b.)

Unrollen Sie die Schleife vollständig. Planen Sie die so resultierenden Schleifenausführungen auf einer hypothetischen CPU mit einer ALU, auf der alle arithmetischen Operationen ausgeführt werden und einer MMU, auf der die Speichertransferoperationen ausgeführt werden. Für die erste Schleifenausführung geben Sie die ungültigen Latenzen für die ALU und die MMU mit an.

## Aufgabe 7.2

Sie haben ein Programm für eine Multi-Core CPU parallelisiert und erwarten, dass die parallelen Programmbestandteile perfekt skalieren. Sie schätzen ferner, dass die parallelen Programmbestandteile 80 % des gesamten Programms ausmachen. Geben Sie die Formel für den maximal erzielbaren Speedup des Gesamtprogramms an und berechnen Sie diesen für eine Multi-Core CPU mit acht Prozessorkernen.

## Aufgabe 7.3

Nennen Sie für den nachfolgenden parallelen Algorithmus das am wenigsten restriktive geteilte Speichermodell einer PRAM, auf der der Algorithmus ausgeführt werden kann. Begründen Sie Ihre Antwort kurz.

---

### Algorithm 1 Paralleler Algorithmus

---

**Eingabe:**

Liste  $T$  ▷ enthält Dreiecke in Fensterkoordinaten  
DepthBuffer ▷ Tiefe für jedes Pixel in Zeichenfläche, initialisiert mit Wert 1

**Ausgabe:**

DepthBuffer ▷ Tiefe für jedes Pixel in Zeichenfläche

```
for all  $v_1, v_2, v_3 \in T$  do in parallel
   $B \leftarrow \text{BOUNDS}(v_1, v_2, v_3)$  ▷ Rechteck um Dreieck
  for  $x = B.min.x$  to  $B.max.x$  do
    for  $y = B.min.y$  to  $B.max.y$  do
      ▷ Mit Funktion  $E()$  wird geprüft, ob Punkt  $x, y$  links von Kante  $v_i, v_j$  liegt
      if  $E(v_1, v_2, x, y)$  and  $E(v_2, v_3, x, y)$  and  $E(v_3, v_1, x, y)$  then
         $D \leftarrow \text{DEPTH}(x, y)$  ▷ Bestimme Tiefe von  $x, y$ 
        if  $D < \text{DepthBuffer}[x][y]$  then
           $\text{DepthBuffer}[x][y] \leftarrow D$ 
        end if
      end if
    end for
  end for
end for
```

---

## Aufgabe 7.4

Es sei das Dreieck  $T$  mit den Eckpunkten in Fensterkoordinaten

$$v_1 = (30, 30)$$

$$v_2 = (60, 60)$$

$$v_3 = (70, 20)$$

und den dazugehörigen Dreieckskanten

$$e_1 = v_1 - v_2$$

$$e_2 = v_2 - v_3$$

$$e_3 = v_3 - v_1$$

gegeben.

**a.)**

Entscheiden Sie mit Hilfe des Scan Konvertierungs-Algorithmus von Pineda aus der Vorlesung, ob es sich bei dem Dreieck um ein Front Face oder ein Back Face handelt. Nehmen Sie an, dass Polygone, deren Eckpunkte gegen den Uhrzeigersinn orientiert sind, Front Faces sind. Geben Sie nicht nur die Lösung, sondern auch den dazugehörigen Rechenweg an.

**b.)**

Bestimmen Sie die baryzentrischen Koordinaten  $b(x, y) = (\lambda_1, \lambda_2, \lambda_3)$  bezüglich  $T$  für den Punkt  $x = (0, 0)$ . Entscheiden Sie *aufgrund dieser Berechnung und mit Begründung*, ob der Punkt innerhalb des Dreiecks liegt.

**c.)**

Werte wie Tiefe, Normalenvektoren oder Farben werden als Vertexattribute mit den Dreiecken gespeichert. Wie kann man aus diesen Vertexattributen Werte für jedes Fragment innerhalb des Dreiecks bestimmen? Welche Rolle spielt der Scan Konvertierungs-Algorithmus von Pineda dabei?

Das Übungsblatt wird am 06.06.2019 besprochen.