**Wilhelm-Schickard-Institut für Informatik**

**Arbeitsbereich Symbolisches Rechnen**
**Prof. Dr. Wolfgang Küchlin**
**Jens Haußmann, M. Sc.**

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

01.03.2018

Wintersemester 2017/2018

# Praktikum Betriebssysteme - Tag 3

## POSIX-Threads – Matrixmultiplikation

**Aufgabe 3.1** *(Threads erzeugen)*

Schreiben Sie mithilfe der POSIX-Threading-Bibliothek ein Programm, welches zur Laufzeit einen neuen Thread erzeugt. Der neue Thread soll eine Existenzmeldung (z. B. `new thread running`) ausgeben und dann terminieren. Der ursprüngliche Prozess soll dabei erst nach dem neuen Thread terminieren.

**Aufgabe 3.2** *(Matrixmultiplikation)*

Implementieren Sie mithilfe von POSIX-Threads eine Funktion `square()`, welche eine beliebige $n \times n$-Matrix $A$, $n < 10$, quadriert.

In einer ersten Version soll jeder Thread einen Eintrag der Matrix $M = A \cdot A$ berechnen. Da dieses Vorgehen für große Matrizen (etwa $n > 4$) viel zu viele Threads erzeugt und deshalb ungeeignet ist, soll in der zweiten Version jeder Thread eine ganze Reihe der Matrix $M$ berechnen. Am besten wird eine (Mehrprozessor-)Maschine ausgenützt, wenn man die beiden Ansätze verbindet. Dazu müssen Sie zuerst testen, wie groß die Dimension der Matrix $A$ ist, bevor Sie entscheiden, welche Menge an Arbeit jeder Thread erhält. Falls der Rechner über $k$ Prozessoren verfügt, wieviele nebeneinander existierende Threads halten Sie für sinnvoll? Von welchen Eigenschaften Ihres Programms machen Sie Ihre Entscheidung abhängig?

*Bemerkung: Ist $A = (a_{i,j})$ eine $n \times n$-Matrix, so ist $A \cdot A = (m_{i,j})$ mit $m_{i,j} = a_{i,1} \cdot a_{1,j} + \cdots + a_{i,n} \cdot a_{n,j}$.*