Scheduler und Fairness[1]

Definition: Scheduler für Komponenten $K_i, i = 1, \ldots, n$

Ein **Scheduler** ist ein *labelled transistion system* (LTS), dh. ein Quadrupel $(S, \Theta_0, T, \Lambda)$

wobei

S	Zustände
Θ_0	Initialzustände
T	eine Transitionsrelation
Λ	Events (Label)

sind mit

$\Theta_0 \subseteq S$
$T \subseteq S \times \Lambda \times S$
$\Lambda = \{(E, i) : \mathbb{P}(K) \times K \mid i \in E\}$

so daß

$$\forall\, \sigma : S;\; E : \mathbb{P}(K) \bullet (E \neq \emptyset \Rightarrow (\exists\, i : E, \sigma' : S \bullet (\sigma, (E, i), \sigma') \in T))$$

Definition: Schwache Fairness

Schwache Fairness liegt vor, wenn jeder Event, der kontinuierlich enabled ist, auch unendlich oft selektiert wird.

Definition: Starke Fairness

Starke Fairness liegt vor, wenn jeder Event, der unendlich häufig verfügbar ist, auch unendlich oft selektiert wird.

Definition: Starke Prozeß-Fairness

Starke Prozeß-Fairness liegt vor, wenn gilt: Akzeptiert ein Prozeß P nach einem Initial-Trace s unendlich oft irgendein Event, so wird auch unendlich oft ein P-Event ausgelöst.

Definition: Lauf

Sei S ein Scheduler für Komponenten $K_i, i = 1, \ldots, n$.

[1] zu den Vorlesungen vom 11.11.96, 14.11.96 und 18.11.96

- Ein **Lauf** (Run) ist eine Sequenz von Selektionen(E, i)[2] $(E_0, i_0), (E_1, i_1), \ldots$ mit $i_j \in E_j$.
- Ein Lauf ist **prüfbar** von dem Scheduler S, wenn gilt:
  $\forall j : \mathbb{N}_0 \, \exists \sigma_j, \sigma_{j+1}, \ldots \bullet (\sigma_j, (E_j, i_j), \sigma_{j+1}) \in T$
- Ein **fairer Lauf** liegt vor, wenn gilt:
  $\forall I : \{1, \ldots, n\} \bullet ((\overset{\infty}{\exists} j : \mathbb{N}_0 \bullet i \in E_j) \Rightarrow (\overset{\infty}{\exists} j : \mathbb{N}_0 \bullet i = e_j))$

Definition: fairer Scheduler, universeller Scheduler

- Ein **fairer Scheduler** ist ein Scheduler, der nur faire Läufe prüfen kann.
- Ein **universeller Scheduler** ist

  a) fair
  b) kann jeden fairen Lauf prüfen.

Theorem:

Sei `SCHED` ein Scheduler für `n` Komponenten, der wie folgt definiert ist:

```
SCHED = rnd?z1 -> ... rnd?zn -> S(z1, ... ,zn)
```

mit

```
S(z1, ..., zn) = ce?E -> (⊓ i: min(E,z1, ..., zn) •
                          -> select!i
                          -> rnd?zi
                          -> S (z1 = if (1 = i OR 1 ∉ E)
                                     then z1
                                     else z1 −1,
                                  ⋮
                                zn = if (1=n OR n ∉ E)
                                     then zn
                                     else zn −1
                                )
                          )
```

[2] wobei E eine Menge von verfügbaren (enabled) Events, und i ein Event.

```
RAND = ⊓ i : ℕ₀ • rnd!i -> RAND
APP = ⊓ E : 𝔽₁(Σ) • ce!E -> APP
min (E,z₁, ..., zₙ) = min {l:E | ∀ j: 1,...,n • z_l ≤ z_j}
```

mit geeigneten Channel-Deklarationen.

SCHED ist ein universeller fairer Scheduler für n Komponenten.