Architecting System Performance; Scheduling

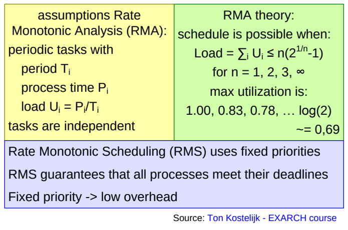
by *Gerrit Muller* TNO-ESI, University College of South East Norway

e-mail: `gaudisite@gmail.com`

`www.gaudisite.nl`

Abstract

Scheduling plays a crucial role in resource allocation to get desired system performance. This document discusses local and global scheduling.



Distribution

June 5, 2018
status: preliminary draft
version: 0

Single Resource Scheduling

Scheduling of time critical operations on a single resource:

- Earliest Deadline First

  optimal

  complex to realize

- Rate Monotonic Scheduling

  no full utilization

  simple to realize

Earliest Deadline First

• Determine deadlines	in Absolute time (CPU cycles or msec, etc.)
• Assign priorities	Process that has the earliest deadline gets the highest priority (no need to look at other processes)
• Constraints	Smart mechanism needed for Real-Time determination of deadlines Pre-emptive scheduling needed

EDF = Earliest Deadline First

Earliest Deadline based scheduling
for (a-)periodic Processing

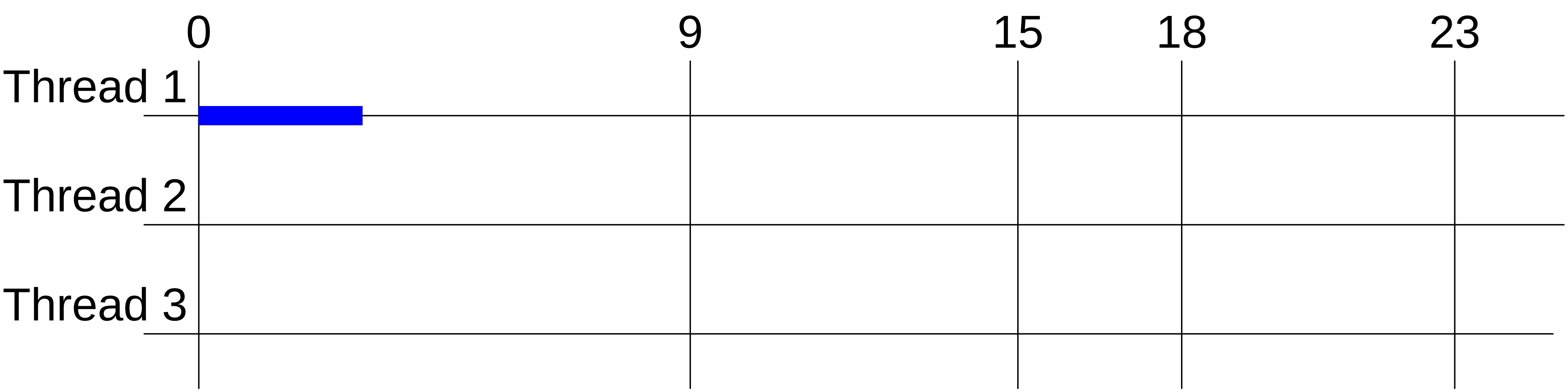
The theoretical limit for any number of processes
is 100% and so the system is schedulable.

Exercise Earliest Deadline First (EDF)

Calculate loads and determine thread activity (EDF)

Thread	Period = deadline	Processing	Load
Thread 1	9	3	33.3%
Thread 2	15	5	
Thread 3	23	5	

Suppose at t=0, all threads are ready to process the arrived trigger.



Source: Ton Kostelijk - EXARCH course

Rate Monotonic Scheduling

• Determine deadlines (period)	in terms of Frequency or Period (1/F)
• Assign priorities	Highest frequency (shortest period) ==> Highest priority
• Constraints	Independent activities Periodic Constant CPU cycle consumption Assumes Pre-emptive scheduling

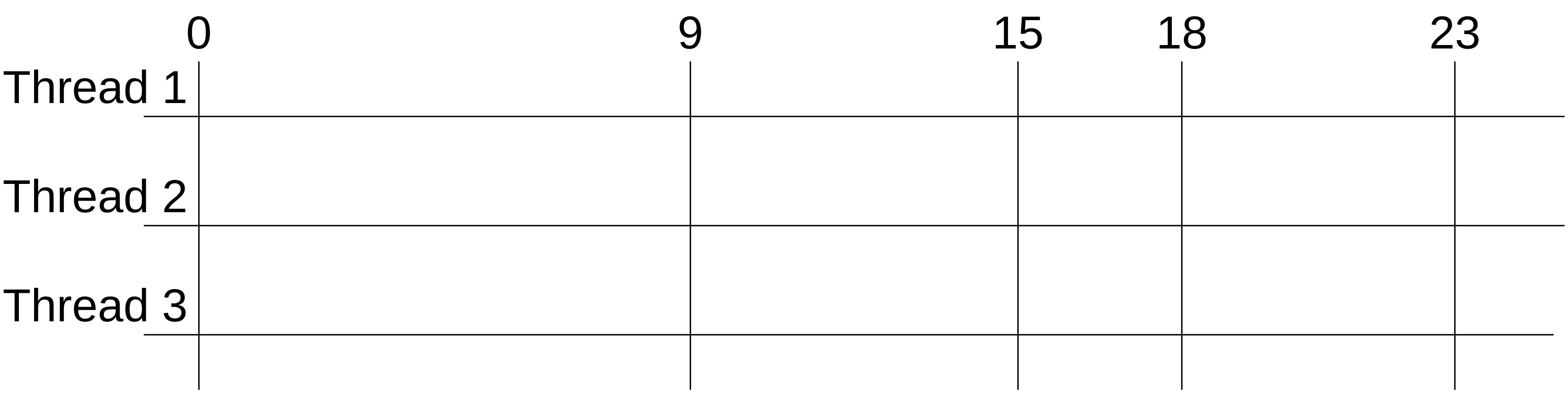
RMS = Rate Monotonic Scheduling

Priority based scheduling for Periodic Processing of tasks with a guaranteed CPU - load

Calculate loads and determine thread activity (RMS)

Thread	Period = deadline	Processing	Load
Thread 1	9	3	33.3%
Thread 2	15	5	
Thread 3	23	5	

Suppose at t=0, all threads are ready to process the arrived trigger.



Source: Ton Kostelijk - EXARCH course

RMS-RMA Theory

assumptions Rate Monotonic Analysis (RMA):	RMA theory:
periodic tasks with period T_i process time P_i load $U_i = P_i/T_i$ tasks are independent	schedule is possible when: $\text{Load} = \sum_i U_i \leq n(2^{1/n}-1)$ for n = 1, 2, 3, ∞ max utilization is: 1.00, 0.83, 0.78, … log(2) ~= 0,69

Rate Monotonic Scheduling (RMS) uses fixed priorities

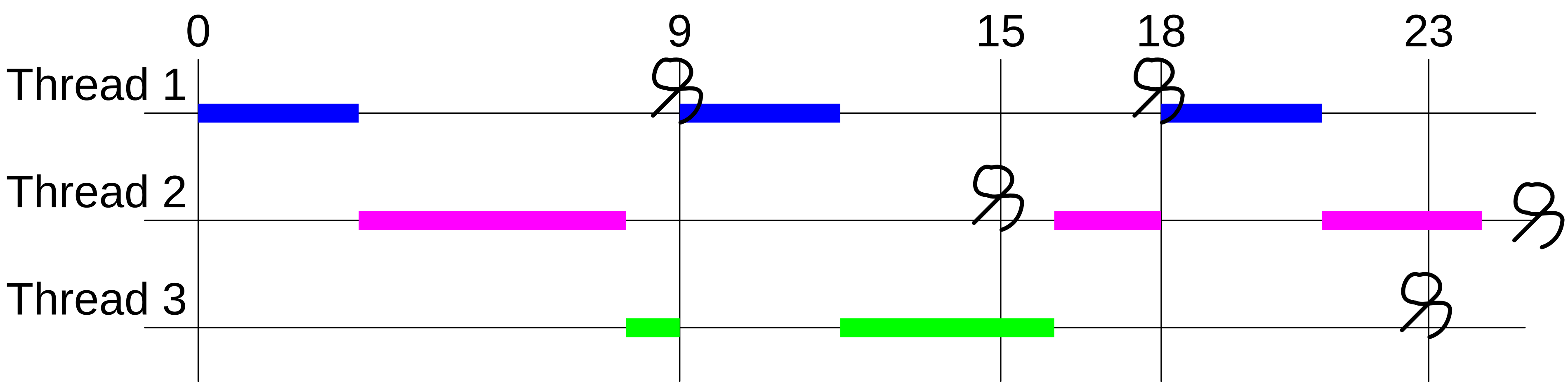
RMS guarantees that all processes meet their deadlines

Fixed priority -> low overhead

Source: Ton Kostelijk - EXARCH course

Answers: loads and thread activity (EDF)

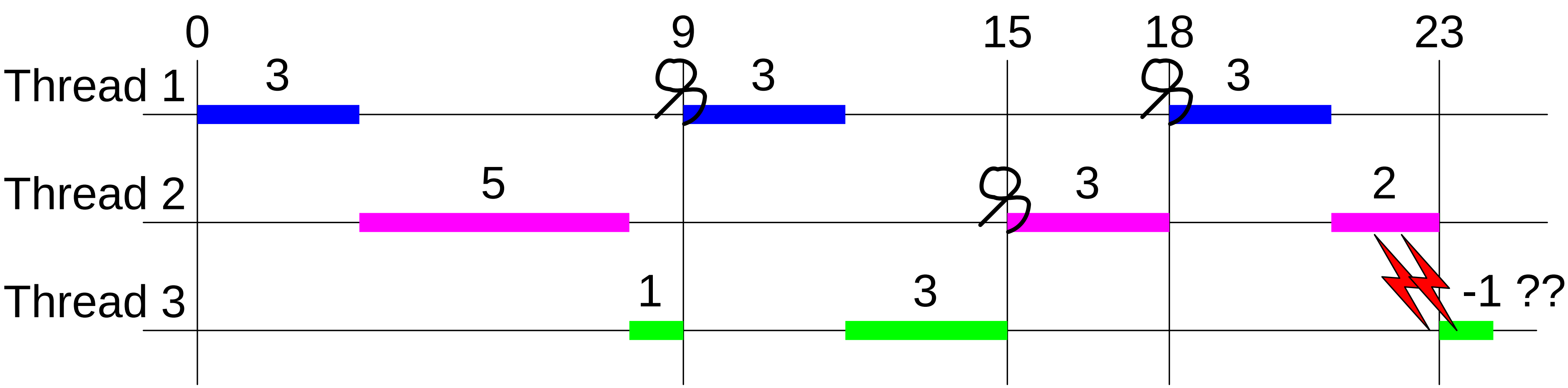
Thread	Period = deadline	Processing	Load
Thread 1	9	3	33.3%
Thread 2	15	5	33.3%
Thread 3	23	5	21.7%
			88.3%



Source: Ton Kostelijk - EXARCH course

Answers: loads and thread activity (RMS)

Thread	Period = deadline	Processing	Load
Thread 1	9	3	33.3%
Thread 2	15	5	33.3%
Thread 3	23	5	21.7%
			88.3%



Source: Ton Kostelijk - EXARCH course

From Simple-Local to Complex-Global

A **perspective** on dynamic behavior is to view the system as set of **periodic behaviors**.

Periodic behavior is easier to **model** and **analyze**, e.g. using RMS and RMA.

Modern systems and Systems of Systems consists of complex **networks of concurrent resources**.

Typically, a combination of more advanced **global** scheduling is combined with simple **local** scheduling.