Graduate Operating Systems

Spring 2023
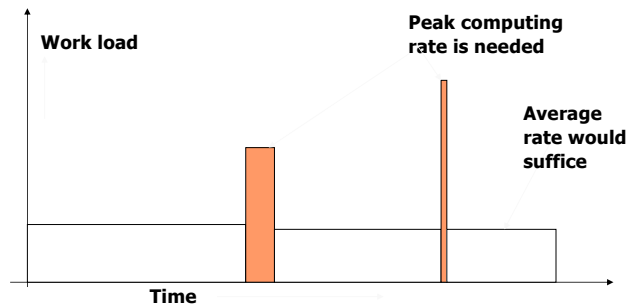
1

Paper "DVS"

- Real-Time Systems
- Dynamic Voltage Scaling (DVS, DFS)
- Over-designed systems (peak performance)
- Periodic task model
- Earliest Deadline First (EDF)
- Rate Monotonic Scheduling (RM)
- Schedulability test

2

Peak vs. Average Performance

Work load

Peak computing
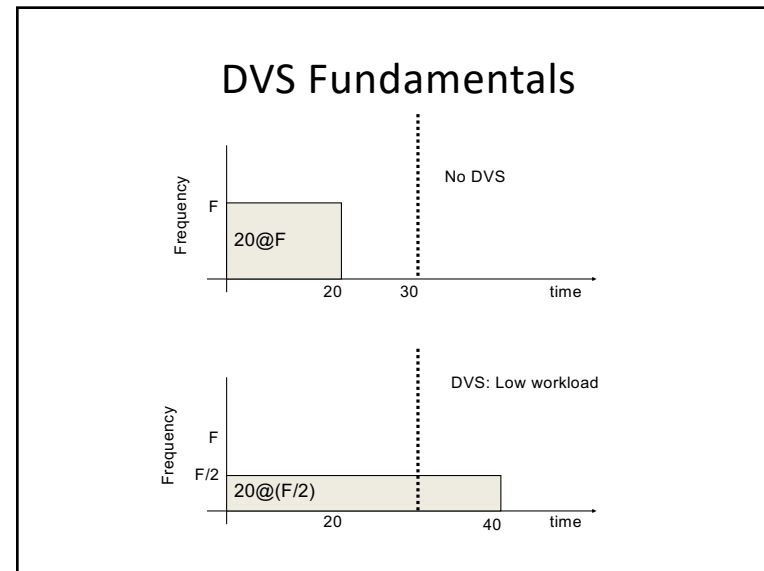rate is needed

Average
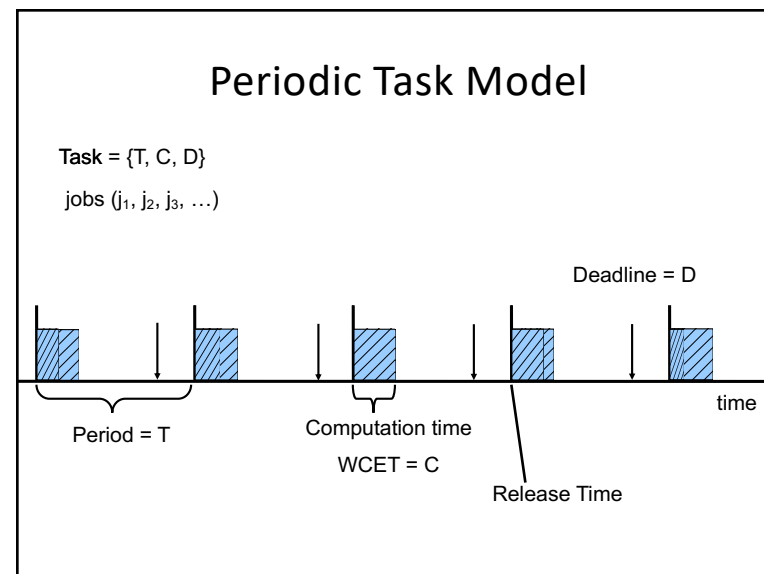rate would
suffice

Time

3

DVS Fundamentals

- Processors are based on CMOS technology where dynamic power is the bottleneck

- Dynamic power (due to switching activity)
  - Power depends on V^2 and f
  - Achievable f depends on V

- Energy = P * $t_{execution}$

4

DVS Fundamentals

Frequency

No DVS

F

20@F

20 30 time

Frequency

DVS: Low workload

F

F/2

20@(F/2)

20 40 time

5

Periodic Task Model

Task = {T, C, D}

jobs (j_1, j_2, j_3, …)

Deadline = D

time

Period = T

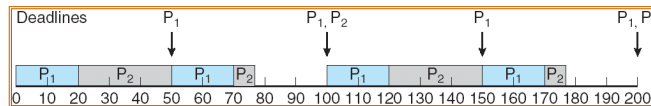Computation time

WCET = C

Release Time
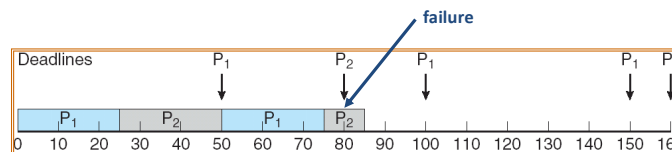
6

RMS (Rate Monotonic Scheduling)

Process P_1: service time = 20, period = 50, deadline = 50

Process P_2: service time = 35, period = 100, deadline = 100

7

Missed Deadlines with RMS

Process P_1: service time = 25, period = 50, deadline = 50
Process P_2: service time = 35, period = 80, deadline = 80

RMS is guaranteed to work if

N = number of processes

sufficient condition

$$u = \sum_{i=1}^{N} \frac{t_i}{p_i} \le N\left(\sqrt[N]{2} - 1\right);$$

$$\lim_{N \to \infty} N\left(\sqrt[N]{2} - 1\right) = \ln 2 \approx 0.693147$$

N	$N\left(\sqrt[N]{2} - 1\right)$
2	0,828427
3	0,779763
4	0,756828
5	0,743491
10	0,717734
20	0,705298

8

4

EDF (Earliest Deadline First)

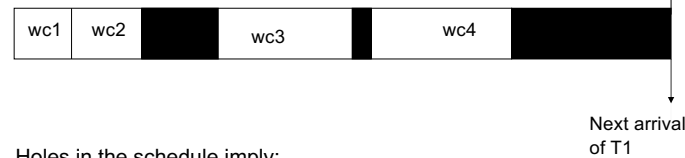Process P_1: service time = 25, period = 50, deadline = 50

Process P_2: service time = 35, period = 80, deadline = 80
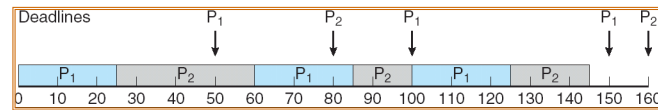
9

Static Voltage Scaling EDF: Motivation

WCi = worst case computation time @ Fmax

Next arrival of T1

Holes in the schedule imply:

EDF Test: $\sum(wc_i/p_i) < 1$ at frequency = F_{max}

10

Static Voltage Scaling EDF

Next arrival of T1

| wc1 | wc2 | | wc3 | | wc4 | |

| K*wc1 | K *wc2 | K * wc3 | K * wc4 |

EDF Test:

$$\sum (wc_i/p_i) < 1 \quad \text{at maximum frequency} = F_{max}$$

Static-VS EDF Test:

$$K* [\sum (wc_i/p_i)] = 1 \quad \text{at frequency} = F_{max}/K$$

11

Static EDF: Example

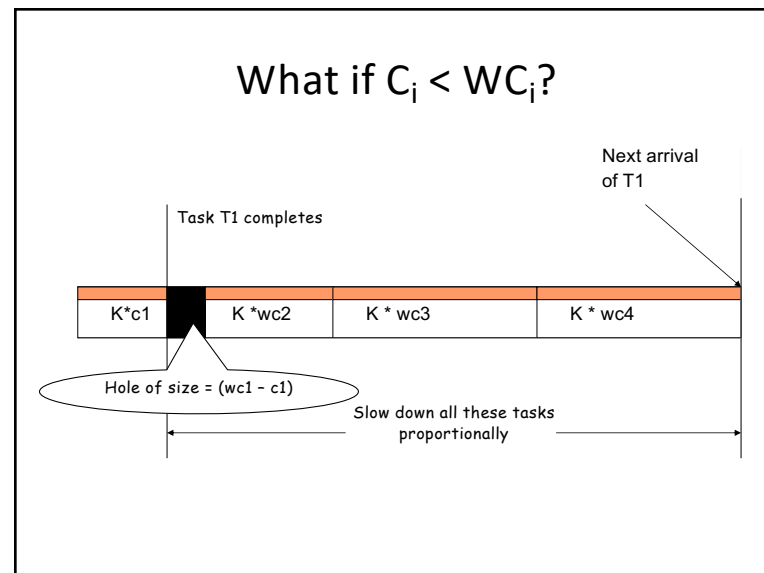| Task	Computing Time	Period
1	3 ms	8 ms
2	3 ms	10 ms
3	1 ms	14 ms

Available frequencies:
1.00, 0.75

Schedulability test for $\alpha = 1.00$

$3/8 + 3/10 + 1/14 \leq 1$ ---> **Return true**

Schedulability test for $\alpha = 0.75$

$3/8 + 3/10 + 1/14 \leq 0.75$ ---> **Return true**

D_{T1}: 16
D_{T2}: 10
D_{T3}: 14

D_{T1}: 16
D_{T2}: 20
D_{T3}: 14

D_{T1}: 16
D_{T2}: 20
D_{T3}: 28

D_{T1}: 24
D_{T2}: 20
D_{T3}: 28

12

6

What if $C_i < WC_i$?

Next arrival
of T1

K*c1	K *c2	K * c3	K * c4

More holes left unexploited

13

What if $C_i < WC_i$?

Next arrival
of T1

Task T1 completes

K*c1	K *wc2	K * wc3	K * wc4

Hole of size = (wc1 – c1)

Slow down all these tasks
proportionally

14

What if $C_i < WC_i$?

Next arrival
of T1

| K*c1 | K' *wc2 | K' * wc3 | K' * wc4 |

CPU Cycles are conserved by slowing down the
remaining tasks

15

Cycle Conserving RT-DVS

- When a task set completes its first release,
  compare real execution time with worst case
  specified initially.

- Any idle time in that period can be used to
  conserve energy.

- Rescale frequency that avoid idle cycles, surplus
  time is used to run other remaining tasks at lower
  frequency.

16

Cycle Conserving EDF: Example

Task set @ (Fmax): T1 = (3,6) and T2 = (6,12)
U = 3/6 + 6/12 = 1 @ (Fmax)

F_m

Frequency

T1

T2

Time

0   1   10

New utilization = 1/6 + 6/12 = 0.67
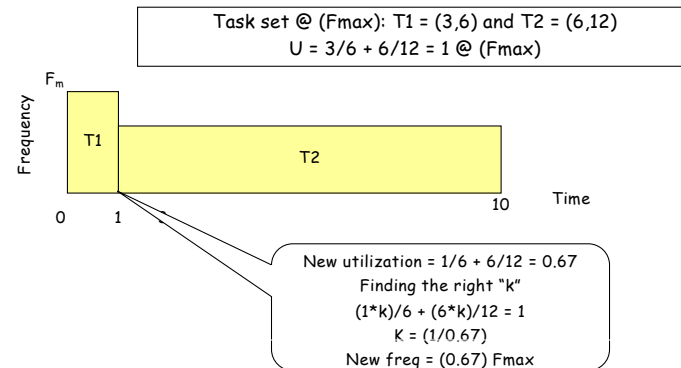Finding the right "k"
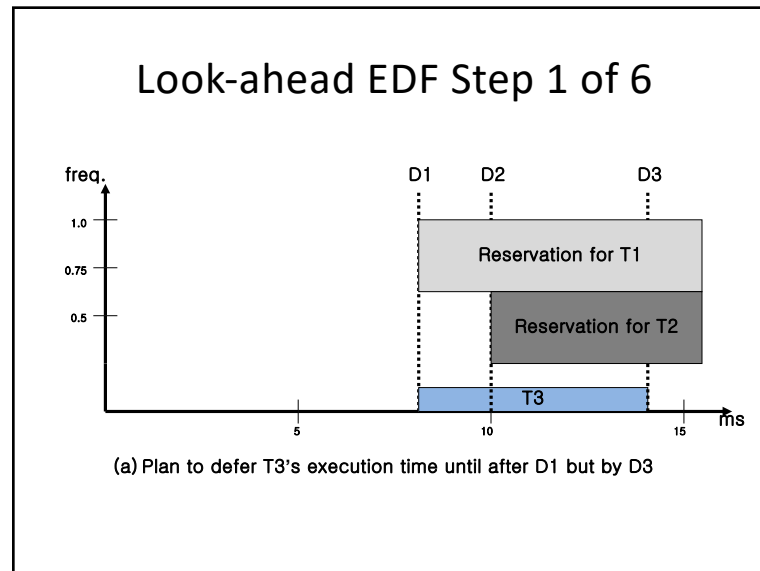(1*k)/6 + (6*k)/12 = 1
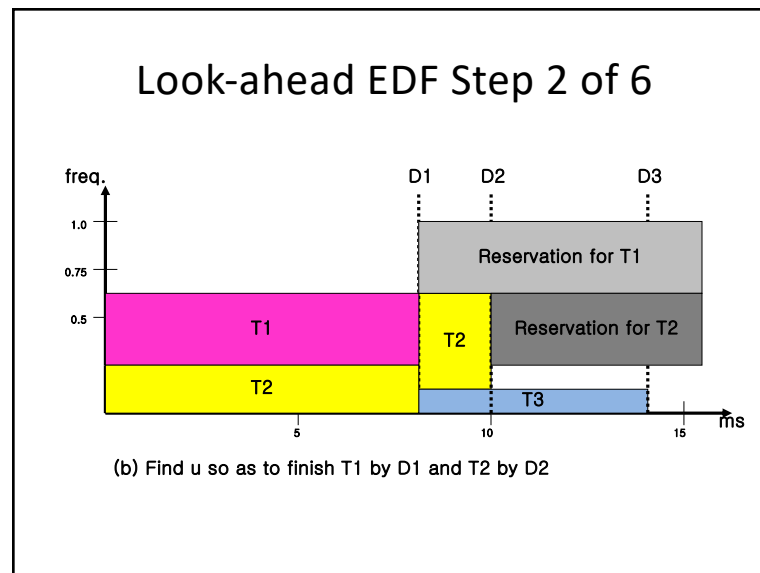K = (1/0.67)
New freq = (0.67) Fmax

17

Look-Ahead EDF

- Defer as much works as possible and set initially to the minimum possible frequency.

- Hence at later stage if a task uses much less than it worst case, deferred work may never be needed.

- It ensures that there are sufficient cycles available for each task to meet its deadline after reserving cycles for higher priority jobs.
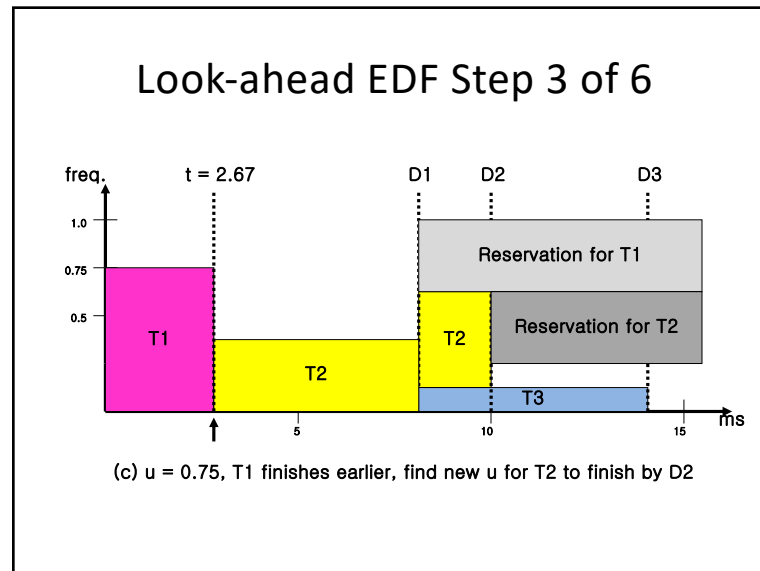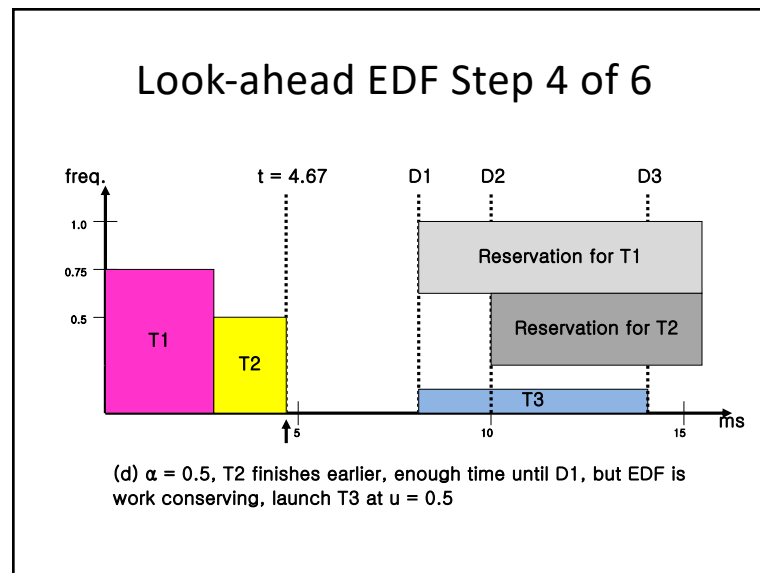
- Best saving of energy.

18

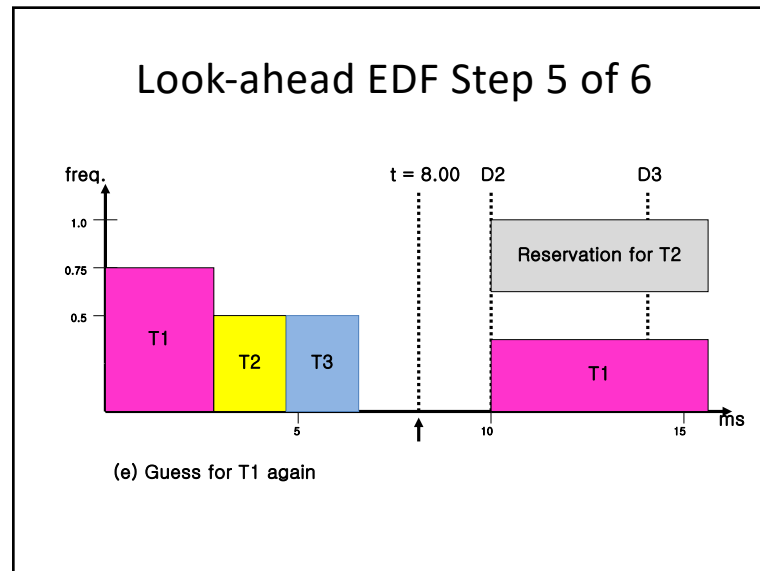## Look-ahead EDF Step 1 of 6

(a) Plan to defer T3's execution time until after D1 but by D3

19

Look-ahead EDF Step 2 of 6

(b) Find u so as to finish T1 by D1 and T2 by D2

20

Look-ahead EDF Step 3 of 6

freq. t = 2.67 D1 D2 D3

Reservation for T1

Reservation for T2

T1 T2 T2 T3

(c) u = 0.75, T1 finishes earlier, find new u for T2 to finish by D2

21

Look-ahead EDF Step 4 of 6

freq. t = 4.67 D1 D2 D3

Reservation for T1

Reservation for T2

T1 T2 T3

(d) α = 0.5, T2 finishes earlier, enough time until D1, but EDF is work conserving, launch T3 at u = 0.5

22

Look-ahead EDF Step 5 of 6

(e) Guess for T1 again

23

Look-ahead EDF Step 6 of 6

(f) u= 0. 5, every task is dynamically scheduled successfully

24

Relative Performance

Energy savings

| Look Ahead EDF |

| Cycle Conserving EDF |

| Static Voltage Scaling EDF |

Schemes

25

Implementation

- Scheduler hooking in the kernel

User Level

| Non–RT DVS | | RT Task Set |

Kernel Level

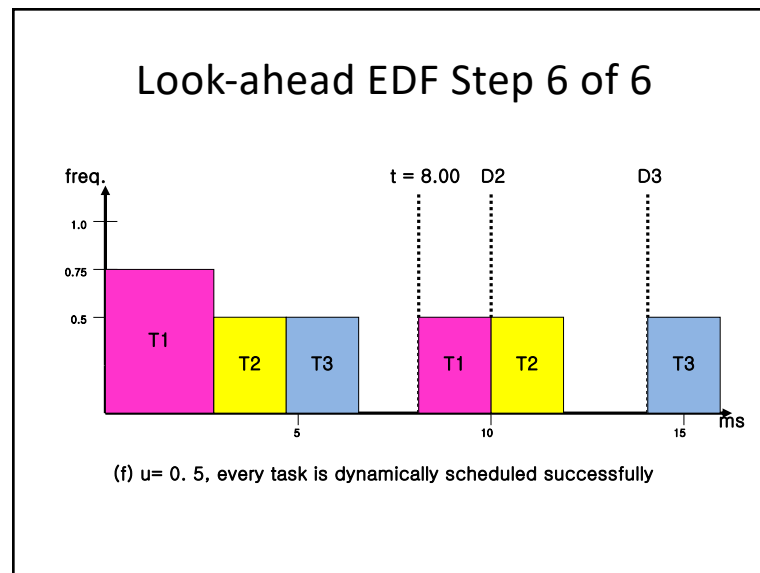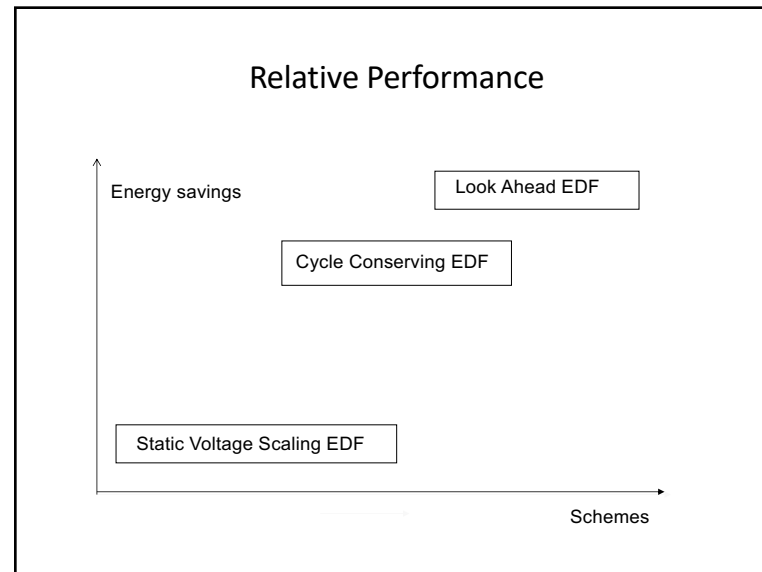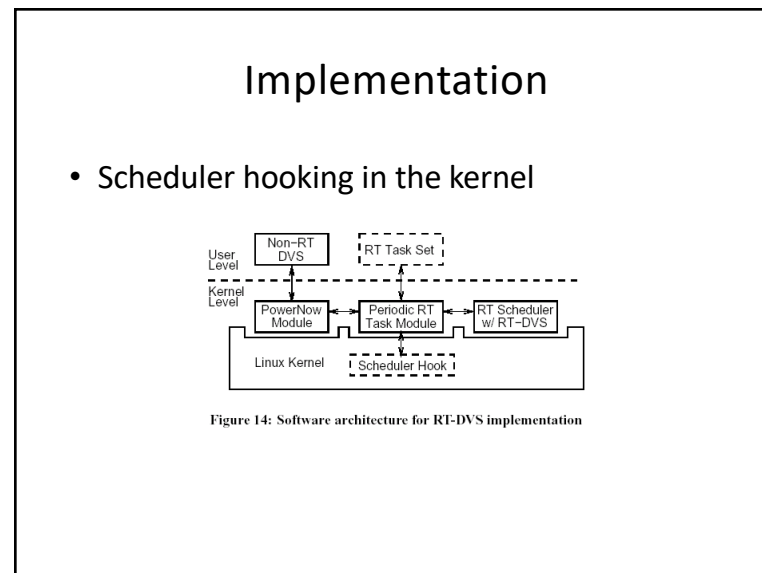| PowerNow Module | | Periodic RT Task Module | | RT Scheduler w/ RT–DVS |

Linux Kernel | Scheduler Hook |

Figure 14: Software architecture for RT-DVS implementation

26