CS528
Energy Efficient Task Scheduling

A Sahu

Dept of CSE, IIT Guwahati

Outline

- Power and Energy Aware
- **Task with Hard Deadlines**
- Energy Efficiency
- Energy Efficient Scheduling
- **Real Time Task System**

Top 500 HPC System

https://www.top500.org/lists/top500/2021/11/

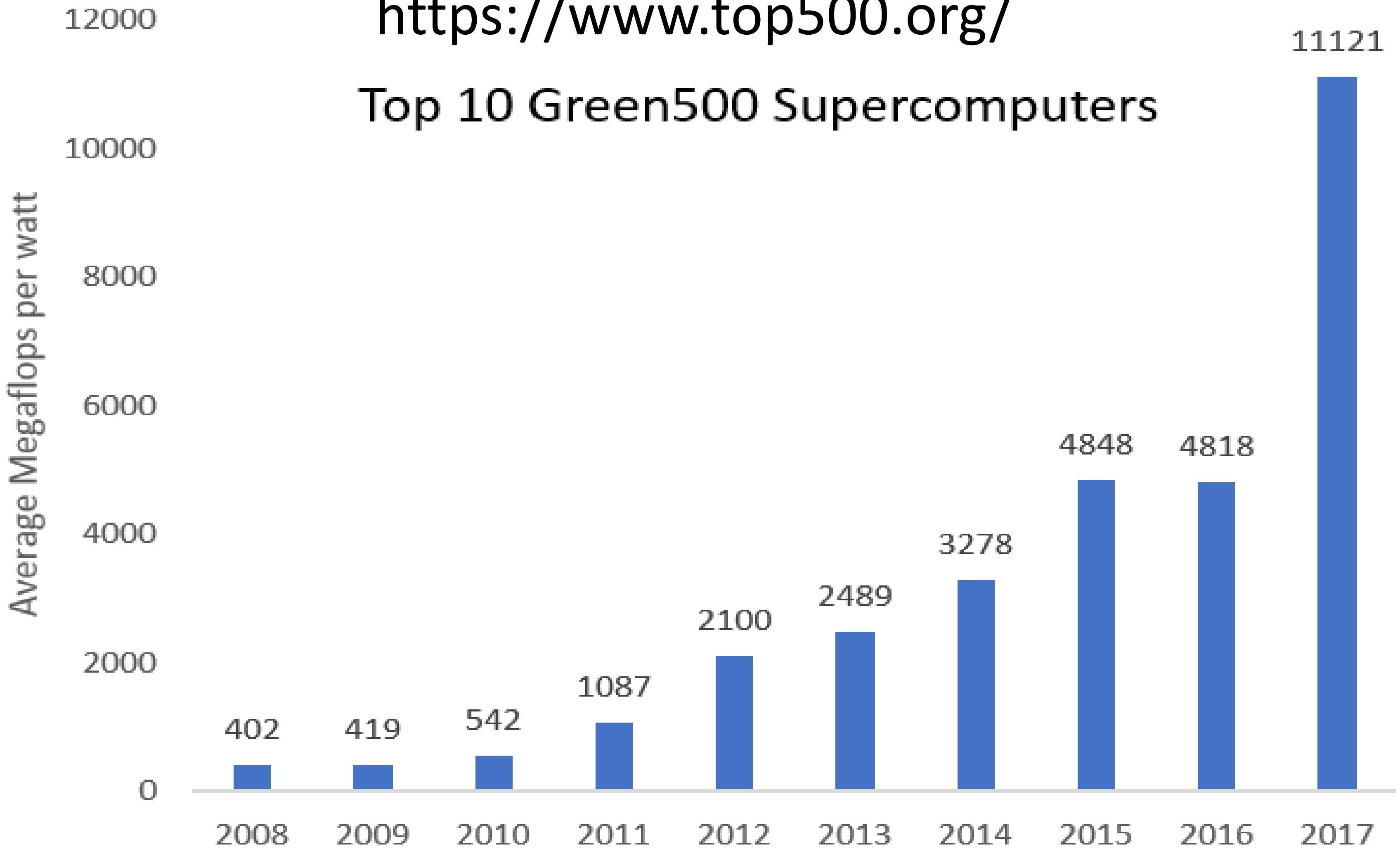
1. Fugaku remains the No. 1 system. Japan
   - 7,630,848 cores : **442 Pflop/s** on HPL Benchmarks.
   - This puts it 3x ahead of the No. 2 system in the list.
2. Summit, an IBM-built system at the Oak Ridge National Laboratory (ORNL), USA,
   - **148.8 Pflop/s** on the HPL benchmark
   - 4,356 nodes, each housing
     - two Power9 CPUs with 22 cores each
     - and six NVIDIA Tesla V100 GPUs, each with 80 streaming multiprocessors (S.M.).
   - Nodes are linked together with a Mellanox dual-rail EDR InfiniBand network.
3. Sierra, at Lawrence Livermore National Lab, USA,
   - Architecture is very similar to Summit.
   - 4,320 nodes with two Power9 CPUs and four NVIDIA Tesla V100 GPUs. Sierra achieved **94.6 Pflop/s.**

Green500: evolution

- 2008: best result = 536 MFlops/Watt
- 2009: best result = 723 MFlops/Watt
  - Cell cluster, ranking 110 in top500
- 2010: best result = 1684 MFlops/Watt
  - IBM BlueGene/Q prototype 1, ranking 101 in top500,
  - Peakperf: 65 TFlops;
- 2011: best result = **2097 MFlops/Watt**
  - IBM BlueGene/Q prototype 2
  - power consumption: 41 kW / Peak 85 TFlop/s

Green500: evolution

https://www.top500.org/



Problems of Energy Efficiency

- Laptop Problem
  - Given the energy budget, maximize number of Job
  - Given the Budget money maximize your satisfaction
    - Go to Restaurant with Rs 100. Choose Items to fill you stomach with your budget.
  - Given Rs 20 for going from IITG to Airport
    - Go to Jhalukbari using IIT G bus freely, Take another public bus pay Rs 20 to reach Airport.
  - Given Rs 10 : not possible, you need to walk…:)
  - Given Rs 600 how to go : Hire Taxi
  - Given Rs 20000 how to go : Hire BMW/Mercedes along with many other cars for security personals

Problems of Energy Efficiency

- Server Problem
  - Budget is not constraints, minimize budget but do all the work (get all the items)
  - I want to Take all item of Thela/Bora..How much I need to pay? ---Bargaining

Server Problem Example : $P_\infty | p_j, d_j | \Sigma E_j$

- We have infinite processors
- Processor can be run at speed f=[0:1], PC=αf^3
- N Tasks with deadlines, Task arrived at time 0, preemption not allowed, p_j at f=1
- Execution time task t_j at freq f $=e_j(t_j,f)=p_j/f$;
- Energy consumption task t_j at freq f $=E*time=PC(f)*e_j(t_j,f)= \alpha f^3 p_j/f = \alpha f^2 p_j$
- **We want to execute all the tasks, and minimize the sum of EC of all the tasks**

Server Problem Example : $P_{\infty}|p_j,d_j|\Sigma E_j$

- We want to execute all the tasks, and minimize the sum of EC of all the tasks
- Solution
  - Select one processor for each of the tasks and total of N processors
  - Run the task at lowest feasible speed to meet the deadline $f_j=p_j/d_j$
- This gives (optimal) minimum ΣE_j
  - Total EC = $\Sigma E_j = \Sigma\, \alpha\, f_j^2\, p_j$
  - As $(a+b)^2 > a^2+b^2$: running two task on one processor with higher speed consume higher energy

Laptop Problem Example : $P_\infty, E_b | p_j, d_j | \Sigma U_j$

- We have infinite processors
- Processor can be run at speed $f=[0:1]$, $PC=\alpha f^3$
- N Tasks with deadlines, Task arrived at time 0, preemption not allowed, p_j at $f=1$
- Execution time task t_j at freq $f = e_j(t_j,f)=p_j/f$;
- Energy consumption task t_j at freq f $=E*time=PC(f)*e_j(t_j,f)= \alpha f^3 p_j/f = \alpha f^2 p_j$
- **We want to execute maximum number of the tasks before deadline given the energy budget**

Laptop Problem Example : $P_\infty, E_b | p_j, d_j | \Sigma U_j$

- **We want to execute maximum number of the tasks before deadline given the energy budget**
- Solution:
  - Sort the tasks based on bare minimum energy requirement $E_j = \alpha f_j^2 p_j$
  - Select the maximum number of task from this set
- Given N item with weight $w_1, w_2, \ldots w_N$: the weight is critical/min energy required of the task
- Select Maximum number of item given the Budget of Knapsack. **0-1 Knapsack Problem**
- NPC and Pseudo polynomial time algorithm exist using Dynamic Programming.

Task with Deadline Vs Real Time Task System

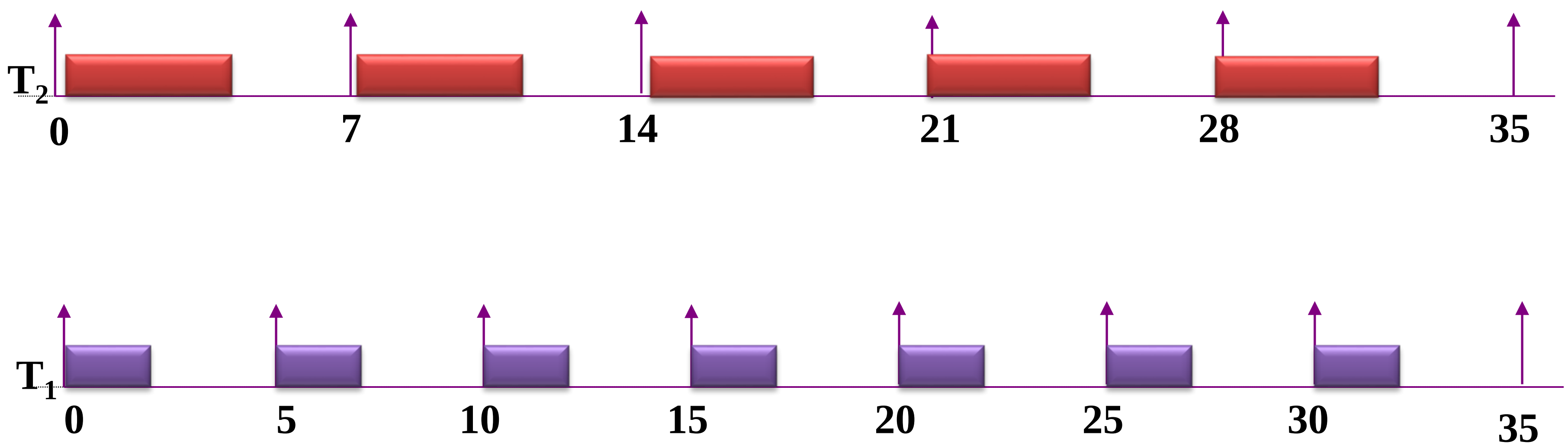
- Task with Deadline: P|pj|ΣU_j
  - Every task have deadline
- Task with Soft Deadline
  - Deadline is not hard, but with QoS or Penalty
    - Airline provide ***free sandwiches to flyer*** when flight get delayed 1 hours/2 hours.
    - More than 3 hours of delay flyer are eligible for free cancellation
- Real time task system: every tasks occurs periodically
  - MP4: (a) video 30 F/S, (b) 16 bits, 2 Channel 44Khz
  - MP4: 1 video task, 2 audio tasks, : repeating one
- Soft Real time task system : Deadline can be soft

Real Time Scheduling

- MPEG, Audio
  - 30 frame/Sec, 50 f/s, 60f/s
- Can you run 4K MKV file on Mobile ?
- Many Periodic Tasks in RT Systems
- **Nice Value** in **Linux**
  - 0-100 for real time task, 101-140 non real time task
  - Size of processor quantum (share) based on nice value

Periodic Task: Real Time Scheduler

- Task with periods : $Ti(c_i,p_i)$ here ci is compute, pi=period
- Each task have to finish before deadline with in the period



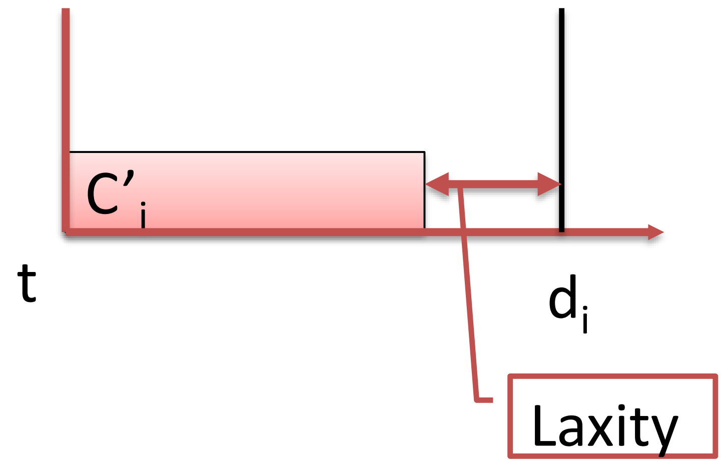
Periodic Tasks

- Necessary schedulability test
  - Sum of utilization factors μ_i must be less than or equal to n, where n is the number of processors
  - **$\mu = \Sigma (c_i / p_i) <= n$**
  - μ_i = Percentage of time the task T_i requires the service of a CPU

Periodic Task: Real Time Scheduler

Assumptions & Definitions

- Tasks are periodic
- No aperiodic or sporadic tasks
- Job (instance) deadline = end of period
- Tasks are preemptable



- Laxity of a Task

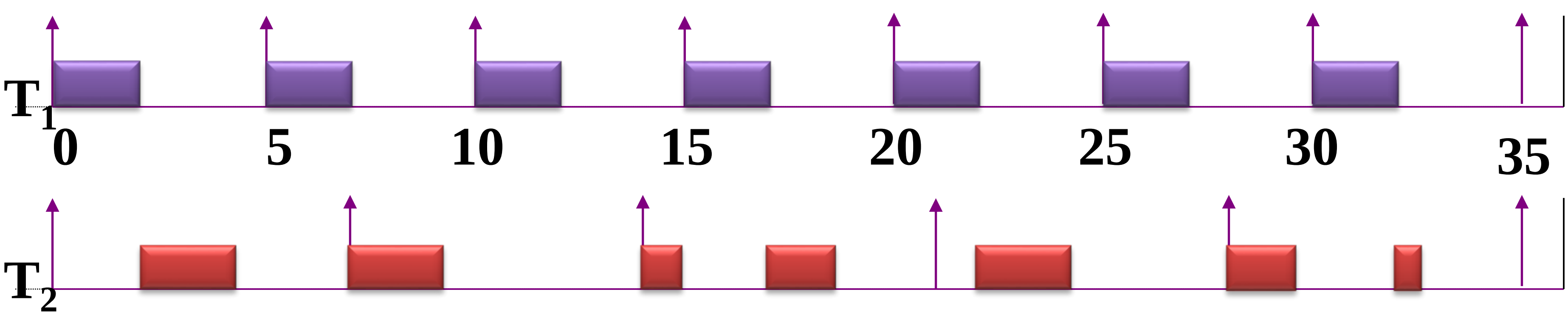
$$\mathbf{T_i = d_i - (t + c_i')}$$

where di: deadline;

t : current time; c_i' : remaining computation time.

Rate Monotonic Scheduling

- **Static Scheduling**
  - Example: Suppose two RT tasks T1 (2,5) and T2(3,7) : Ti(c_i,p_i) here ci is compute, pi=period
  - Task with the smallest period is assigned the highest priority. At any time, the highest priority task is executed.



Rate Monotonic (RM) Scheduling

- **Schedulability check (off-line)**

  - A set of ***n* tasks** is schedulable on a uniprocessor by the RMS algorithm if the processor utilization (utilization test):

$$\sum_{i=1}^{n} \frac{c_i}{p_i} \le n(2^{1/n} - 1)$$

The term $n(2^{1/n} - 1)$ approaches $\ln 2$, (≈ 0.69 as $n \to \infty$).

Earliest Deadline First (EDF)

- **Dynamic Scheduling**
- Task with the smallest deadline/laxity is assigned the highest priority. EDF or **Least Laxity First (LLF)**
  – At any time, the highest priority task is executed.
- **Schedulability check (off-line)**
  – A set of ***n* tasks** is schedulable on a uniprocessor by the EDF algorithm if the processor utilization.

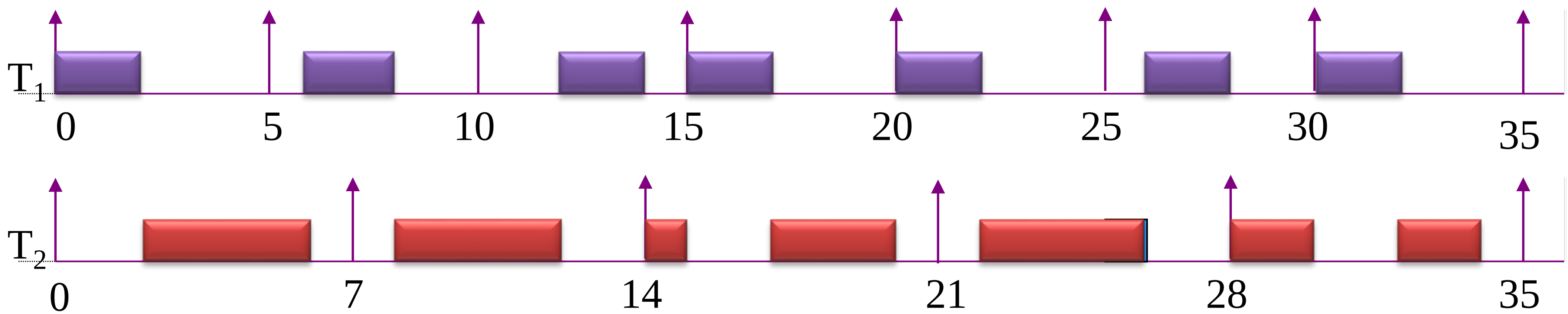
$$\sum_{i=1}^{n} \frac{c_i}{p_i} \leq 1$$

- This condition is both necessary and sufficient.

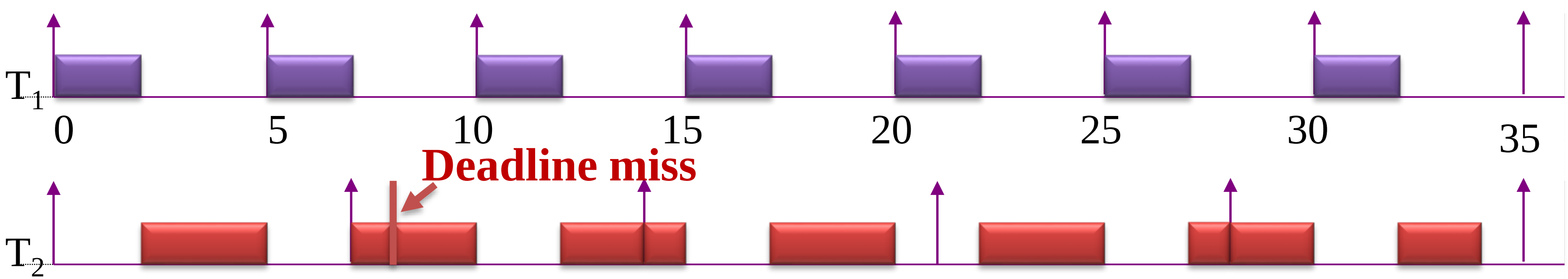
RM & EDF -- Example

Process	Period, T	WCET, C
T_1	5	2
T_2	7	4

EDF schedule



RMS schedule



RT task: energy minimization

- Given a system of ***n*** periodic tasks $T=\{\tau_1, \tau 2, .. \tau_n\}$ and one ***Dynamic Volt-Freq Scaling Processor***
- With $F=\{0, f_1, f_2, f_3, \ldots f_{max}\}$ finite number of freq
- And $f_i < f_{i+1}$.
  - Assume the task system satisfy $\sum(wc_i/p_i) < 1$ **at** f_{max}, wc_i=worst case compute time of i^{th} task
  - It ensure all the period task are schedulable without missing deadline if we run processor at f_{max}

RT task: energy minimization

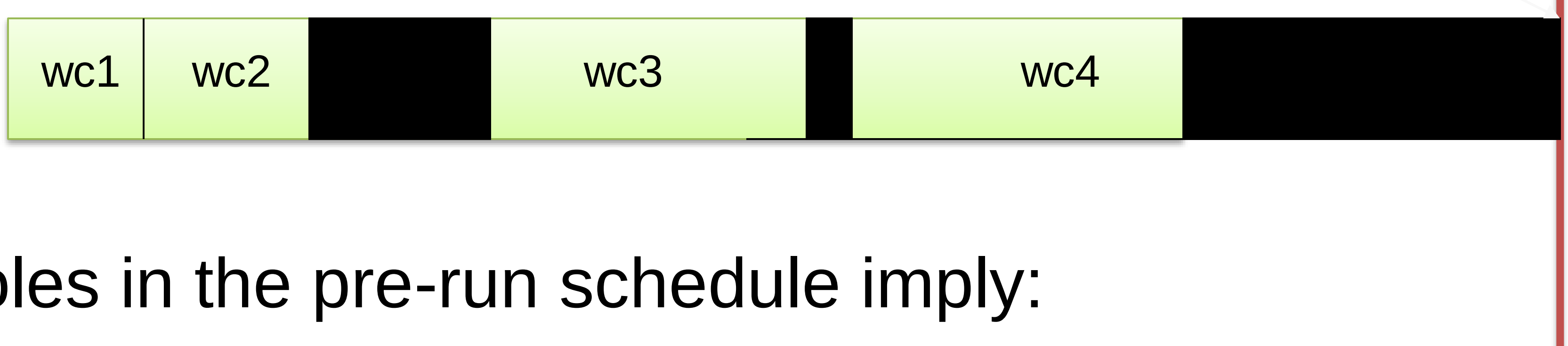
- Design an efficient and elegant way to reduce the power/energy consumption ($E=f^3t$)
  - Hidden assumption: With out missing deadline of any task
- Number of processor is 1
- You may assume: all periodic tasks arrive at time 0
- Deadline of task is period of task

Frequency Scaling EDF: Motivation

Pre-run schedule with holes

WC_i = worst case computation time @ F_{max}

Next arrival of T1



Holes in the pre-run schedule imply:

EDF Test:

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

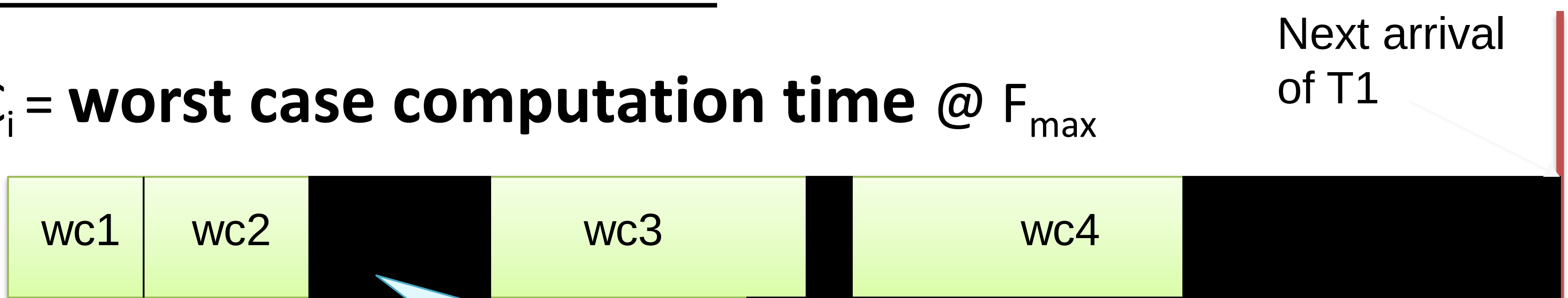
In other words, whenever

$\sum(wc_i/p_i) < 1$ there are holes in the EDF schedule

Frequency Scaling EDF: exploiting holes

Pre-run schedule with holes

WC_i = **worst case computation time** @ F_{max}



Processor typically idles during holes. Instead, the holes can be exploited to slowdown the processor to save energy

How to do it ?
You need design an efficient and elegant way to reduce the Energy Consumption ?