Australian National University

C06

Scheduling

Scheduling Criteria

Algorithms

Temporal Scopes

Purpose of Scheduling

- Two scenarios for scheduling algorithms:

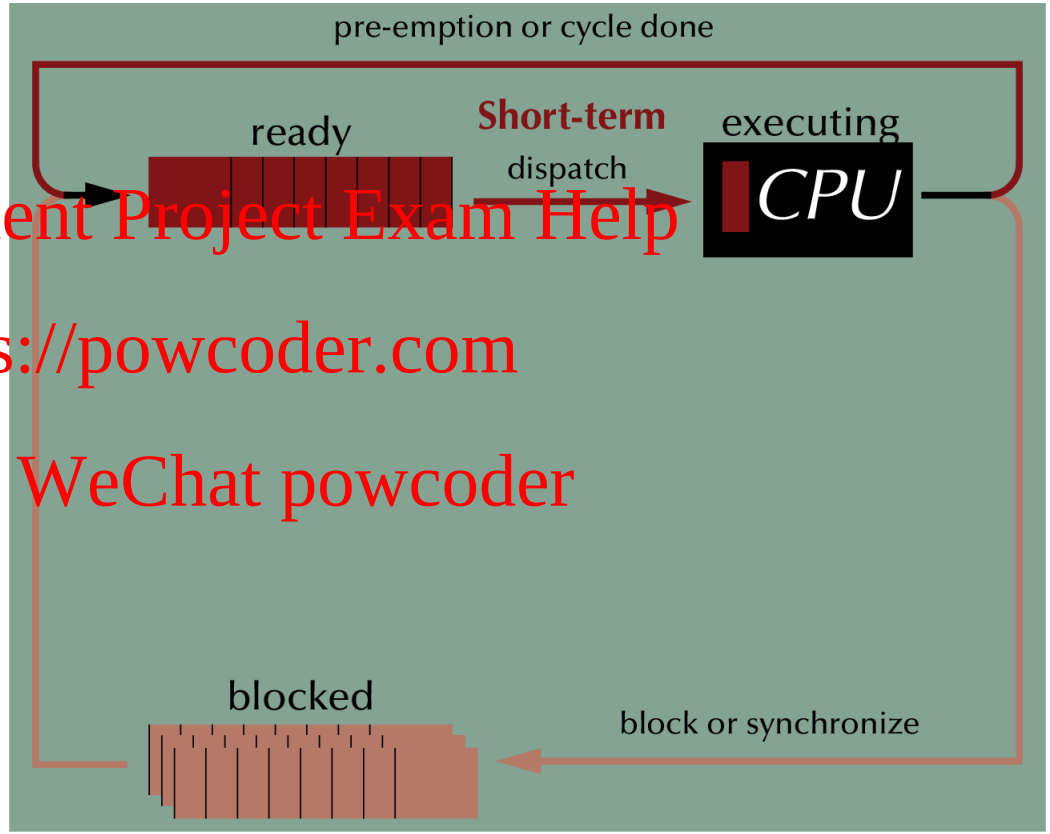
1. Ordering resource assignments (CPU time, network access, …)
   - live, on-line application of scheduling algorithms
2. Predicting system behaviours under anticipated loads
   - simulated, off-line application of scheduling algorithms

- Predictions are used:
  - at compile time: to confirm the feasibility of the system, or to predict resource needs, …
  - at run time: to admit new requests or for load-balancing, …

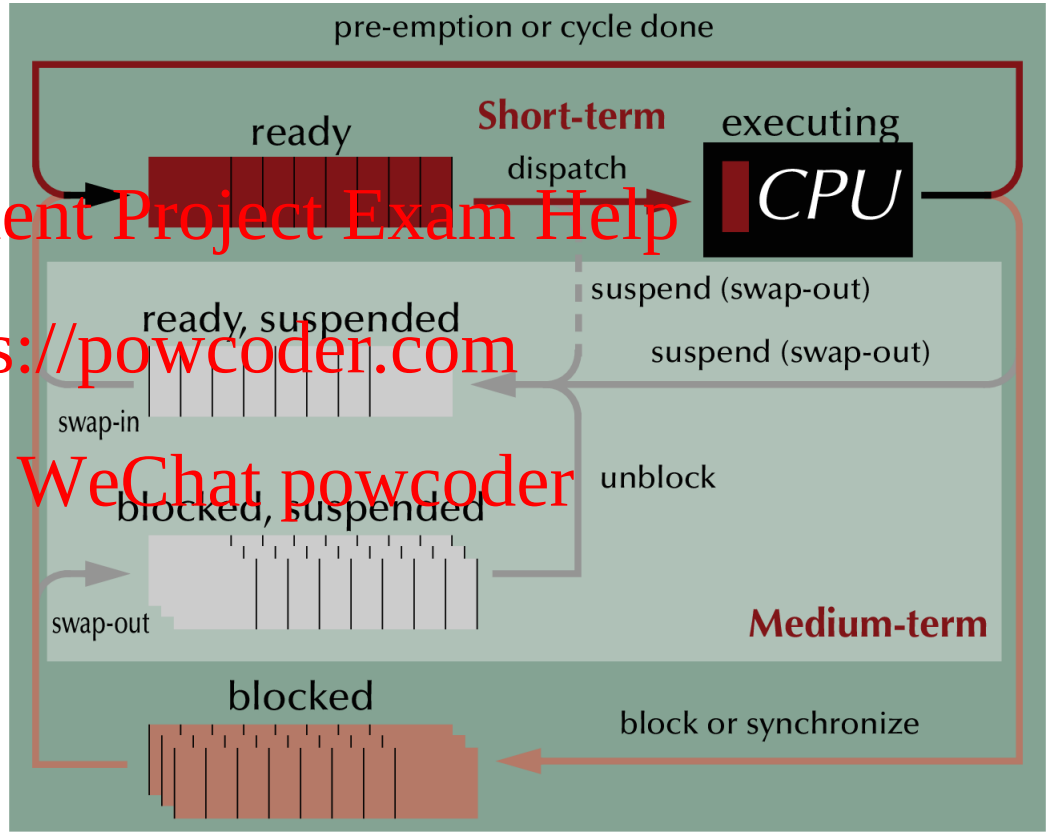
Scheduling Criteria

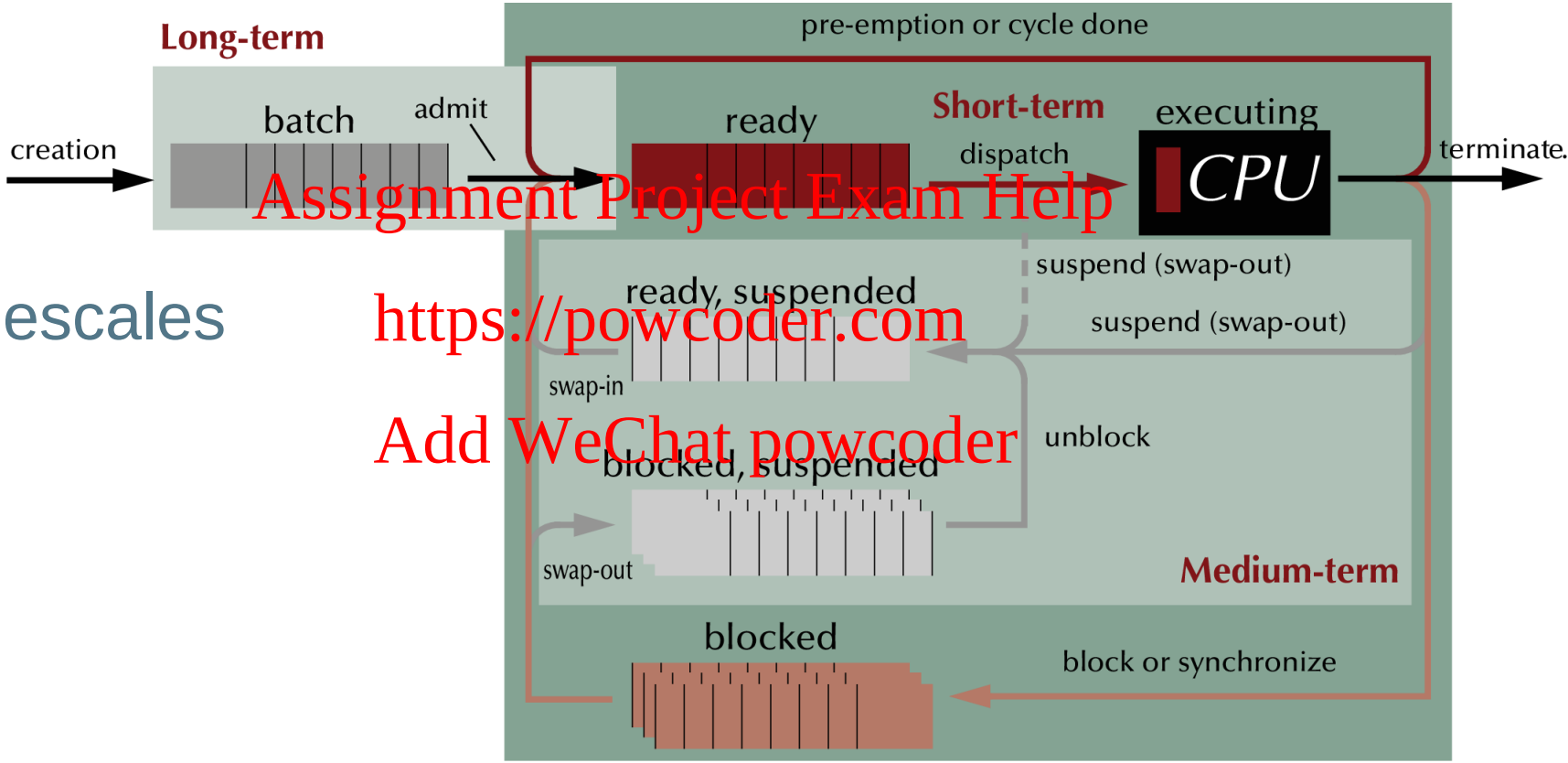
	Performance	Predictability
Process / user perspective: minimize…		
Waiting time	minima / maxima / average / variance	value / minima / maxima
Response time	minima / maxima / average / variance	value / minima / maxima / deadlines
Turnaround time	minima / maxima / average / variance	value / minima / maxima / deadlines
System perspective: minimize…		
Throughput	minima / maxima / average	
Utilization	CPU busy time	

Timescales



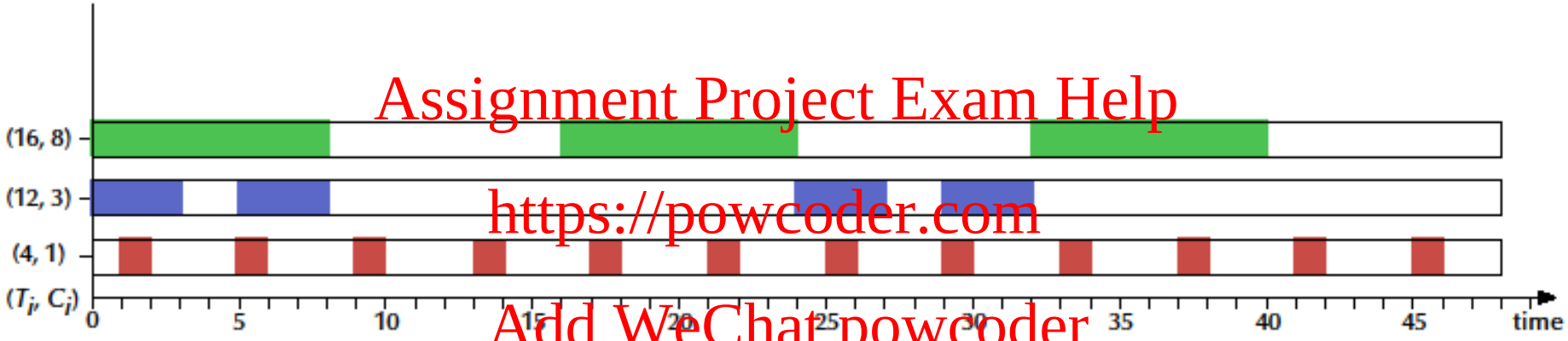
Timescales





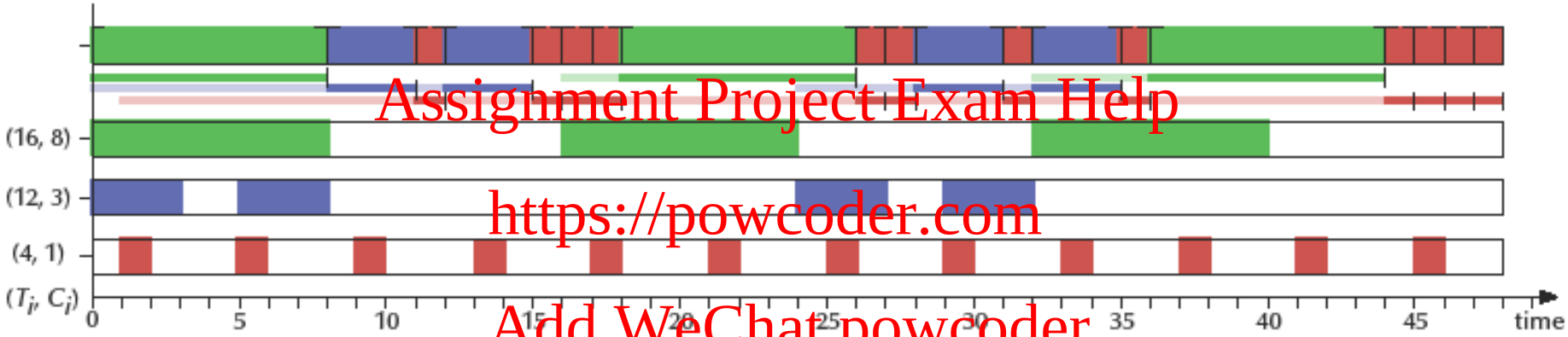
Timescales

Requested Resource Times



- Tasks have an average time between instantiations of T_i and a constant computation time of C_i

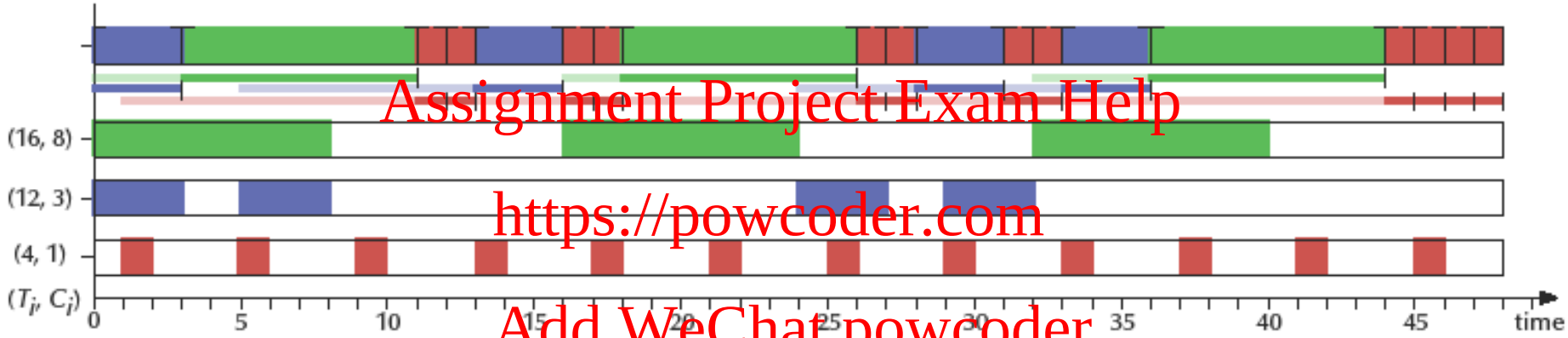
First-Come, First-Served (FCFS)



Waiting time: 0..11, average: 5.9 – Turnaround time: 3..12, average: 8.4

- As tasks apply concurrently for resources, the actual sequence of arrival is non-deterministic
  - hence even a deterministic scheduling schema like FCFS can lead to different outcomes

First-Come, First-Served (FCFS)



Waiting time: 0..11, average: 5.4 – Turnaround time: 3..12, average: 8.0

Shortest possible maximal turnaround time! In this example:

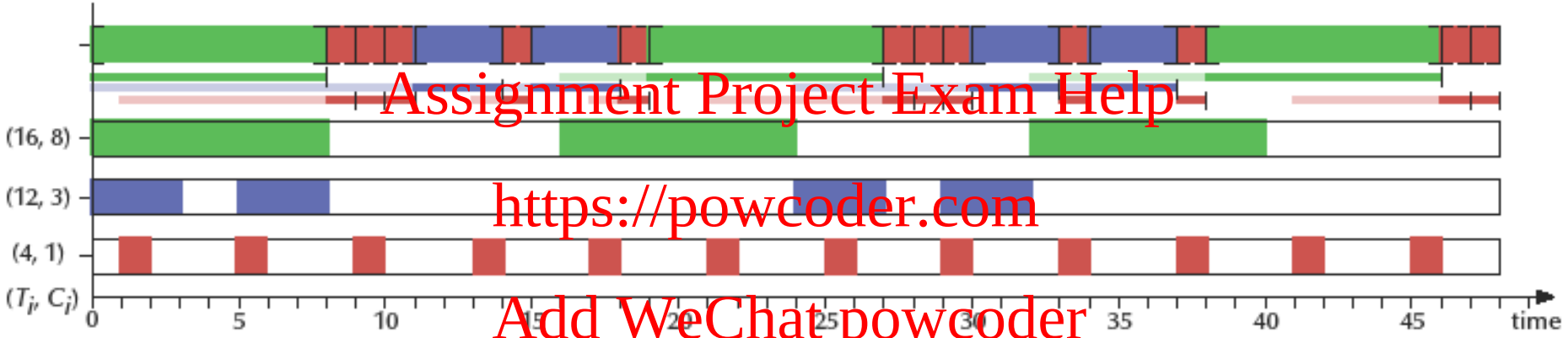
- average waiting times vary between 5.4 and 5.9
- average turnaround times vary between 8.0 and 8.4

Round Robin

Waiting time: 0..5, average: 1.2 – Turnaround time: 1..20, average: 5.8

- Optimized for initial response time
- Extends turnaround time for long tasks
- Bounded maximal waiting time! (depends only on the number of tasks)

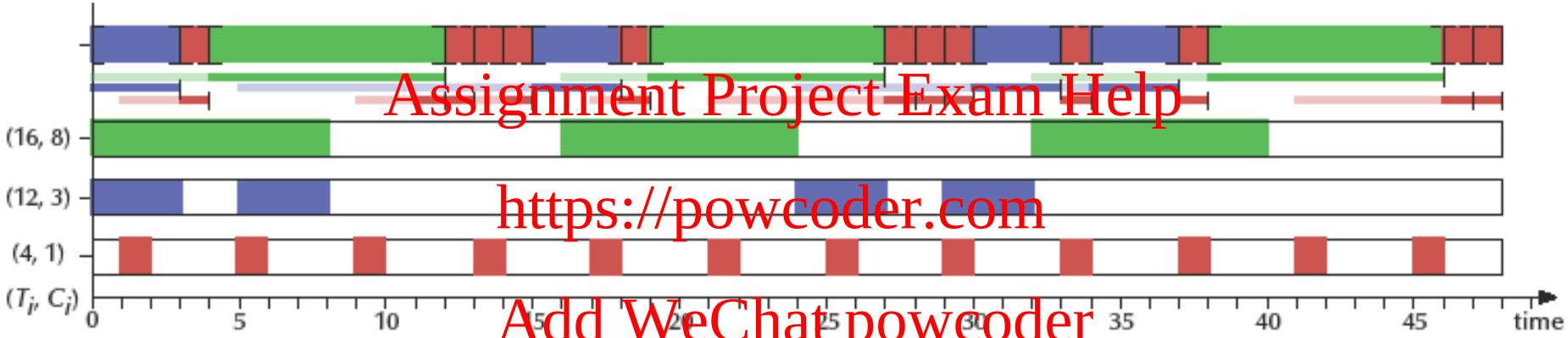
Shortest Job First



Waiting time: 0..11, average: 3.7 – Turnaround time: 1..14, average: 6.3

- Minimizes average turnaround time with minimal task-switches
- Prefers short tasks but all tasks will be handled
- Good if computation times are known and task switches are expensive!

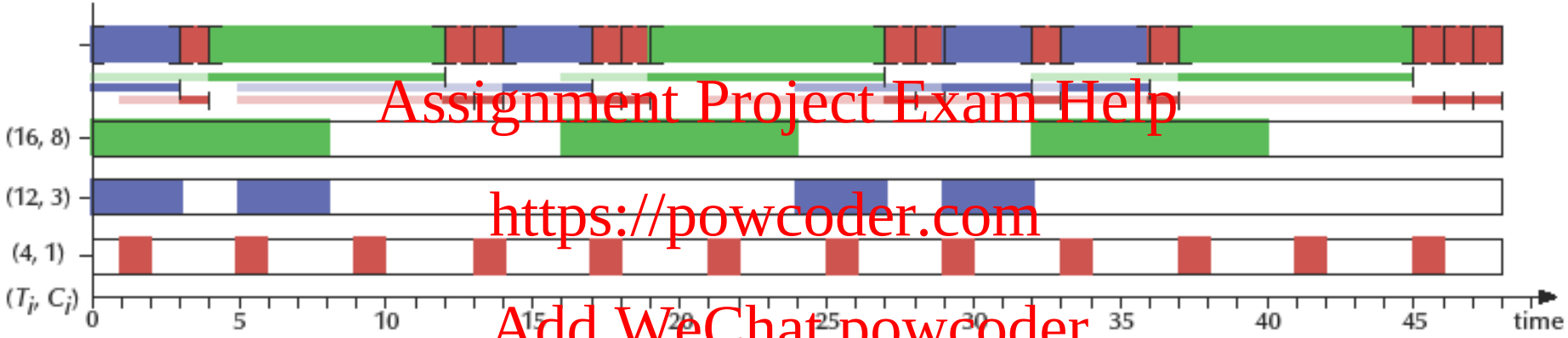
Shortest Job First



Waiting time: 0..10, average: 3.4 – Turnaround time: 1..14, average: 6.0

- Can be sensitive to non-deterministic arrival sequences

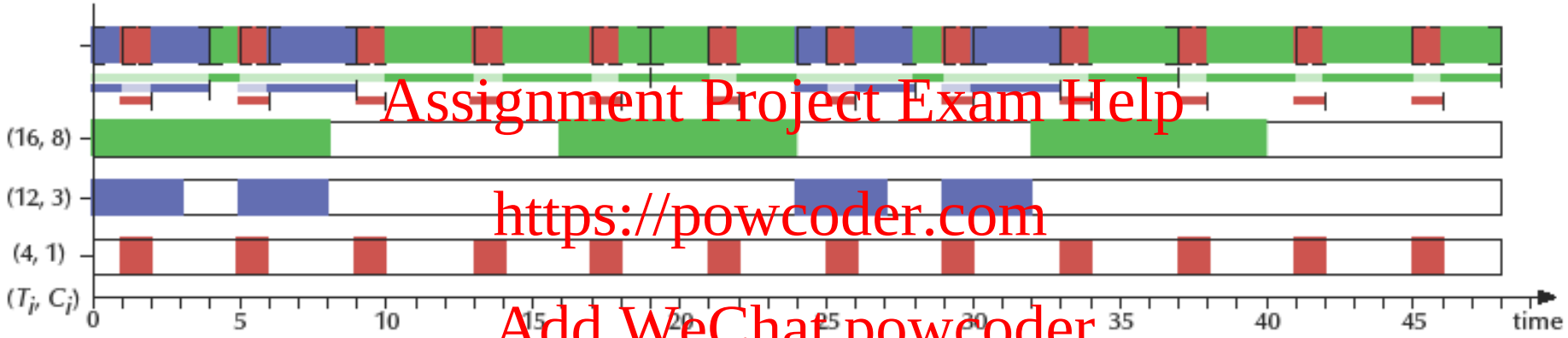
Highest Response Ratio First



Waiting time: 0..9, average: 4.1 – Turnaround time: 2..13, average: 6.6

- Blend between Shortest Job First and First Come First Served: $\frac{W_i+C_i}{C_i}$
- Prefers short tasks but long tasks gain preference over time
- More task switches and worse average than SJF; better upper bound!

Shortest Remaining Time First

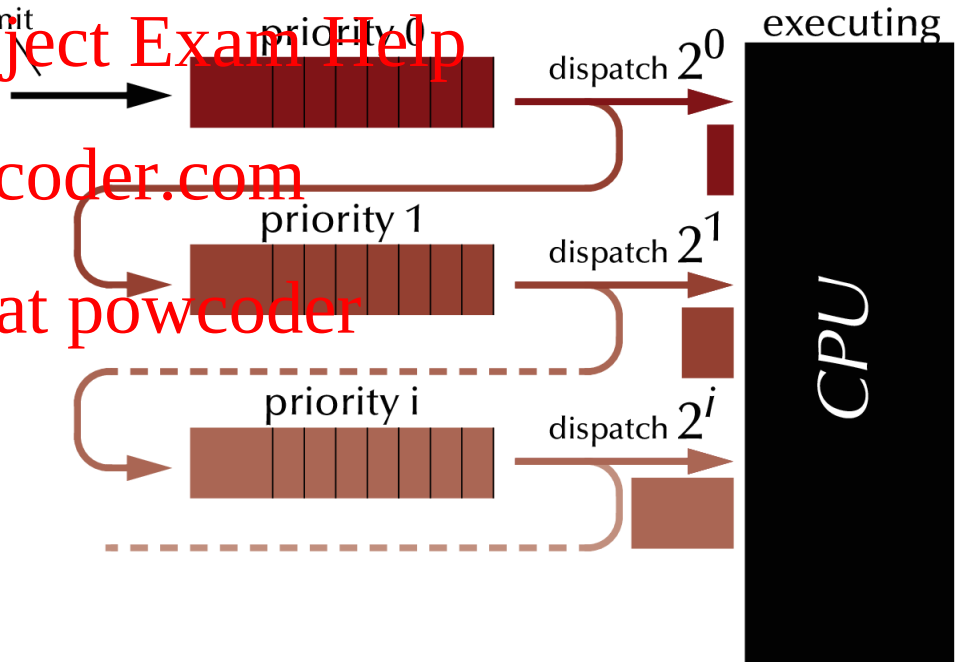


Waiting time: 0..6, average: 0.7 – Turnaround time: 1..21, average: 4.4

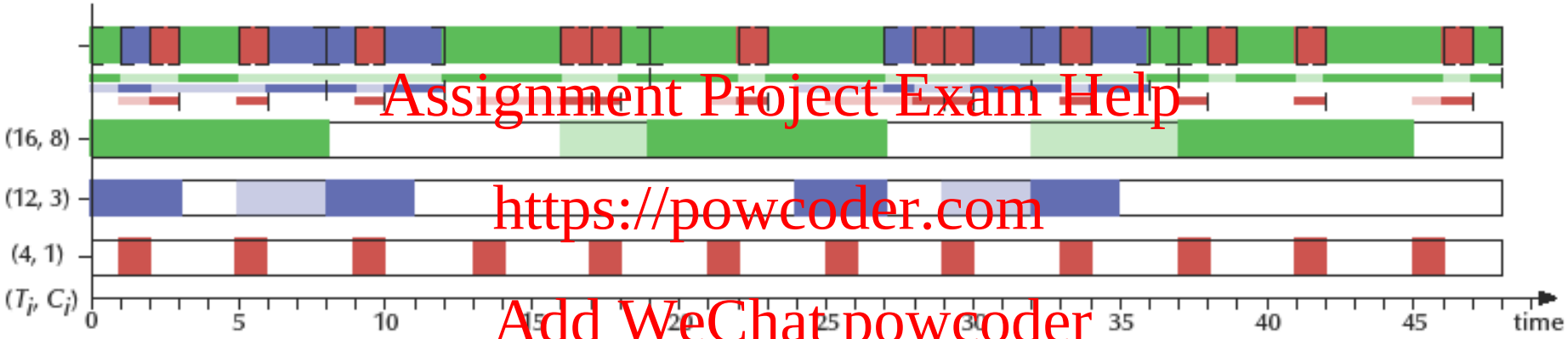
- Optimized for good averages
- Prefers short tasks and long tasks can suffer starvation

Multi-Level Feedback Queue

- Implement multiple hierarchical ready-queues
  - Fetch processes from the highest filled ready queue
  - Dispatch more CPU time for lower priorities (2^i units)
- Processes on lower ranks may suffer starvation
- New and short tasks will be preferred



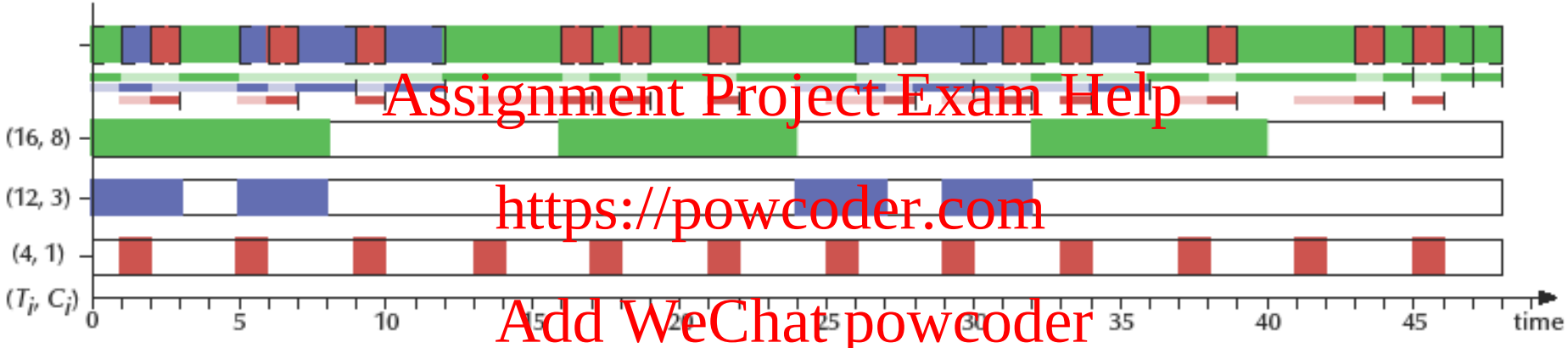
Multi-Level Feedback Queue - Sequential



Waiting time: 0..5, average: 1.5 – Turnaround time: 1..21, average: 5.7

- Optimized for swift initial responses
- Prefers short tasks and long tasks can suffer starvation
- Very short initial response times! and good average turnaround times

Multi-Level Feedback Queue - Overlapping

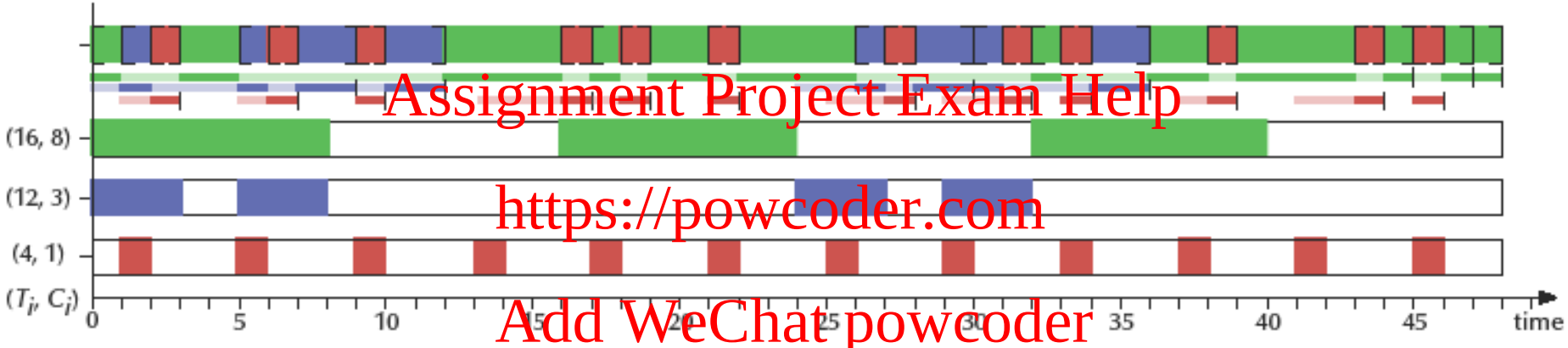


Waiting time: 0..3, average: 0.9 – Turnaround time: 1..45, average: 7.7

Optimized for swift initial responses

- Prefers short tasks and long tasks can suffer starvation
- Long tasks are delayed until all queues run empty!

Multi-Level Feedback Queue - Overlapping

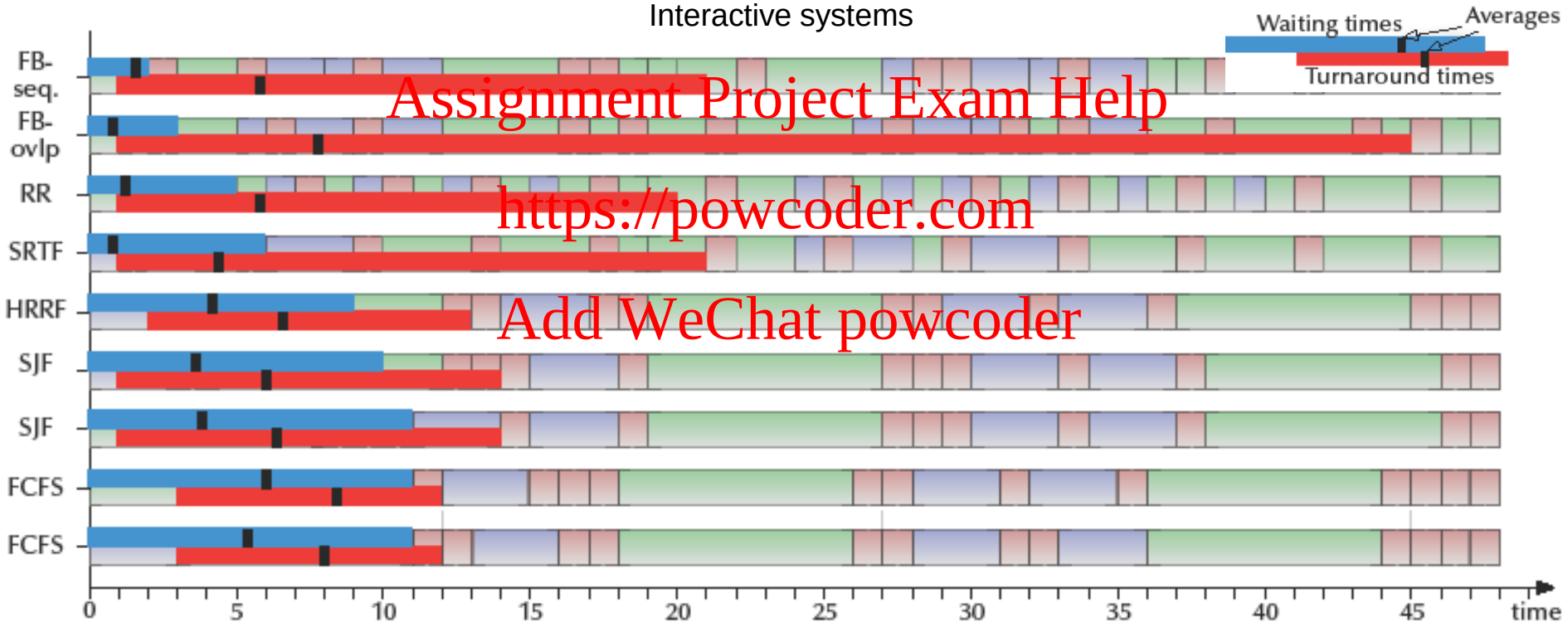


Waiting time: 0..3, average: 0.9 – Turnaround time: 1..45, average: 7.7

Optimized for swift initial responses

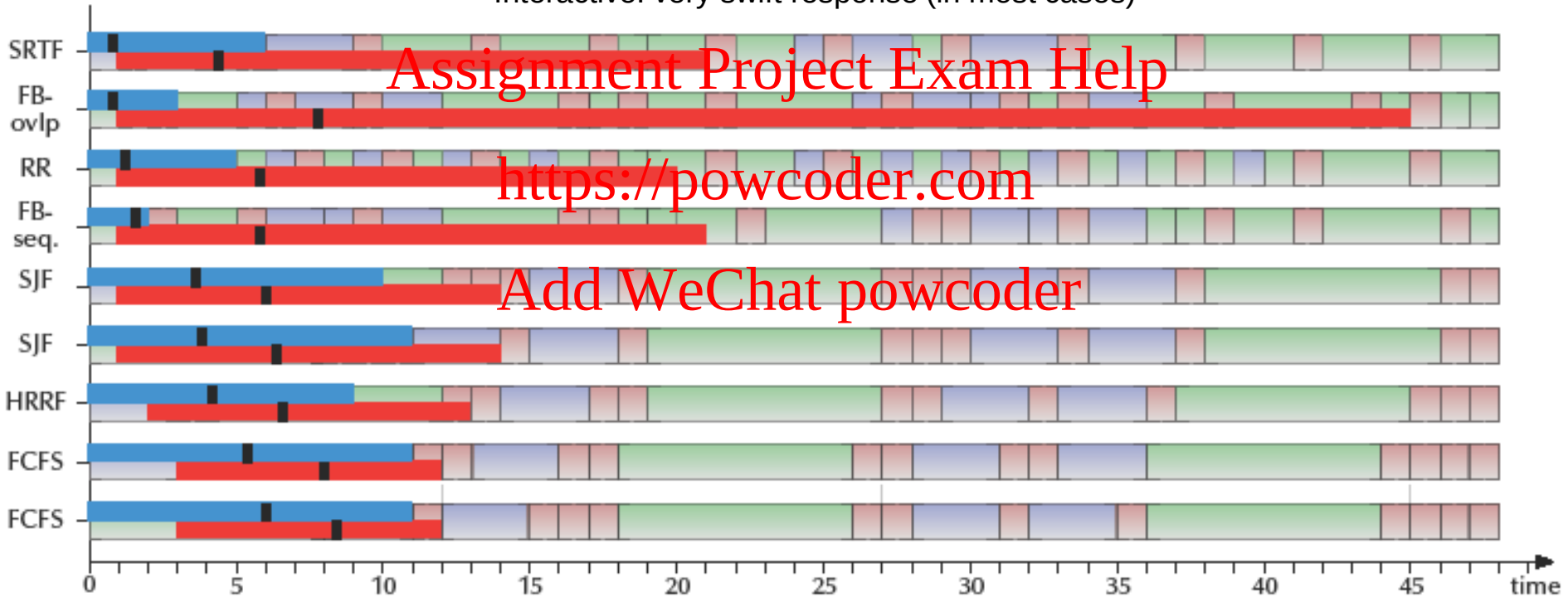
- Prefers short tasks and long tasks can suffer starvation
- Long tasks are delayed until all queues run empty!

Comparison by Shortest Maximum Wait

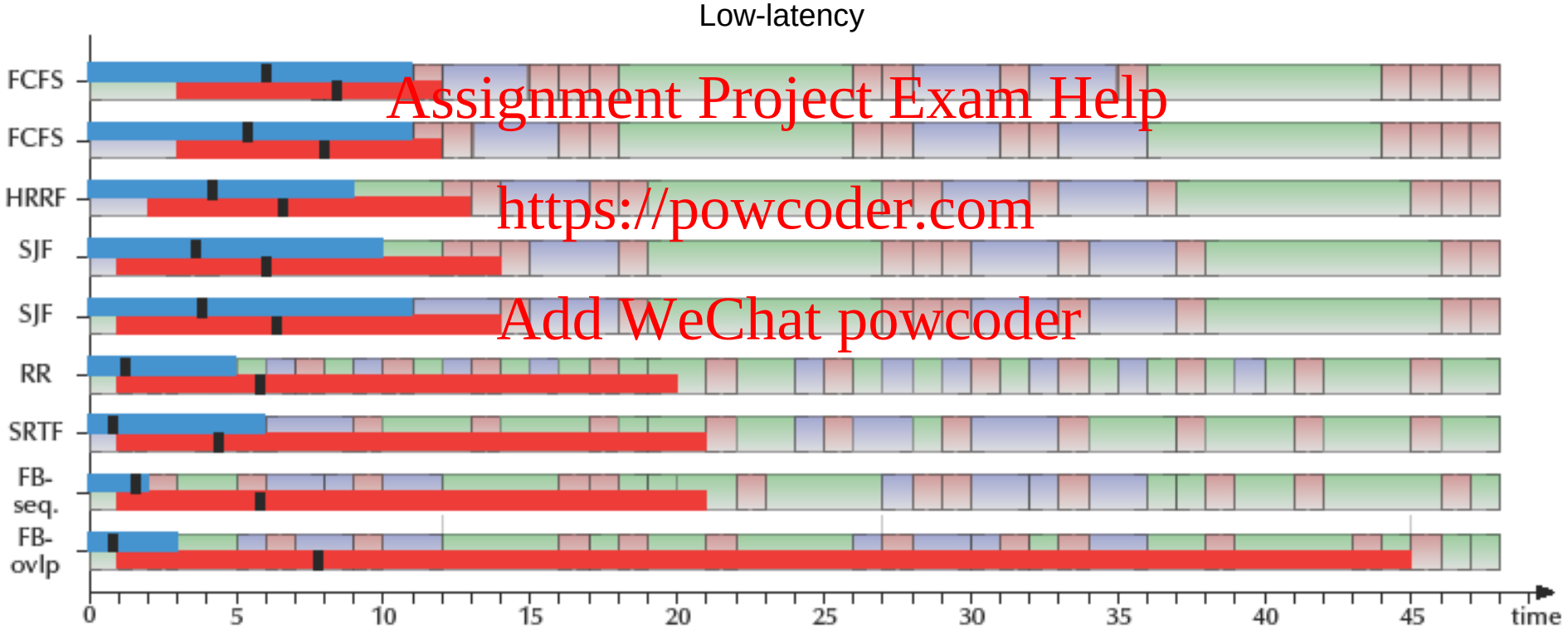


Comparison by Shortest Average Wait

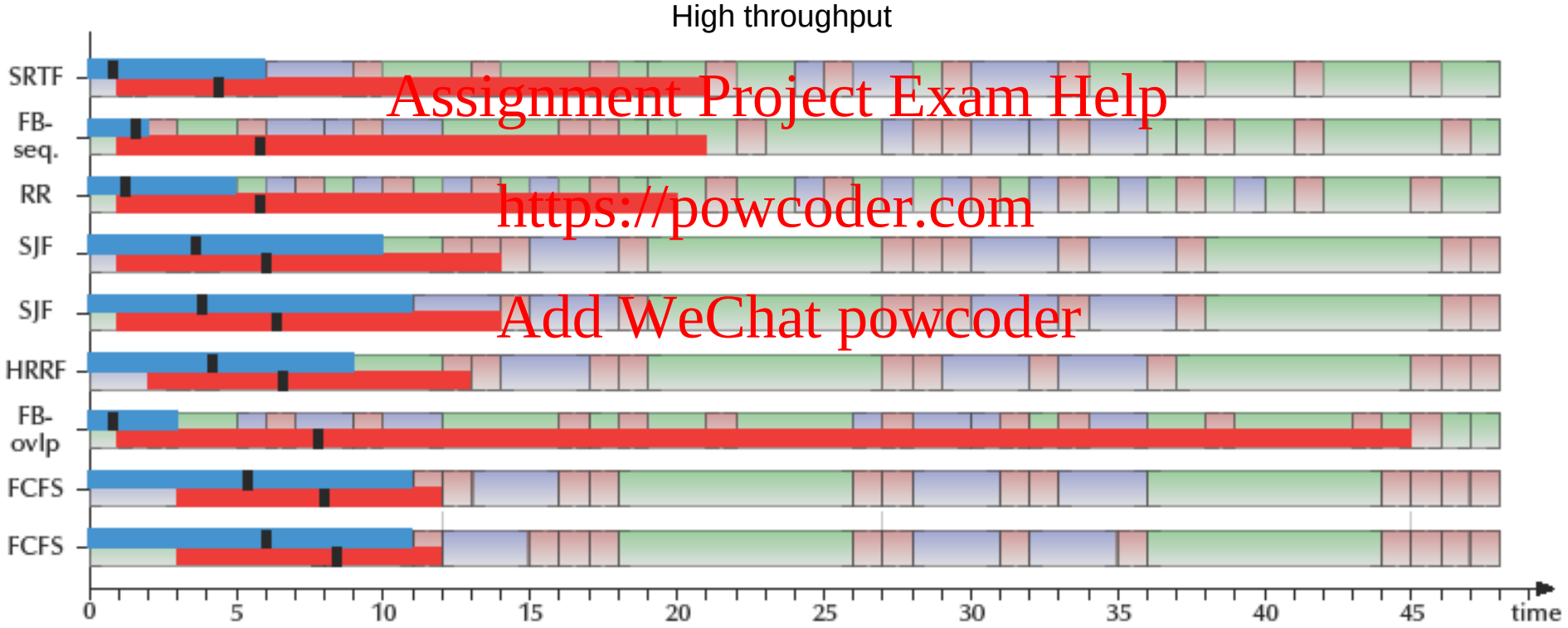
Interactive: very swift response (in most cases)



Comparison by Shortest Maximum Turnaround



Comparison by Shortest Average Turnaround



Comparison: Summary

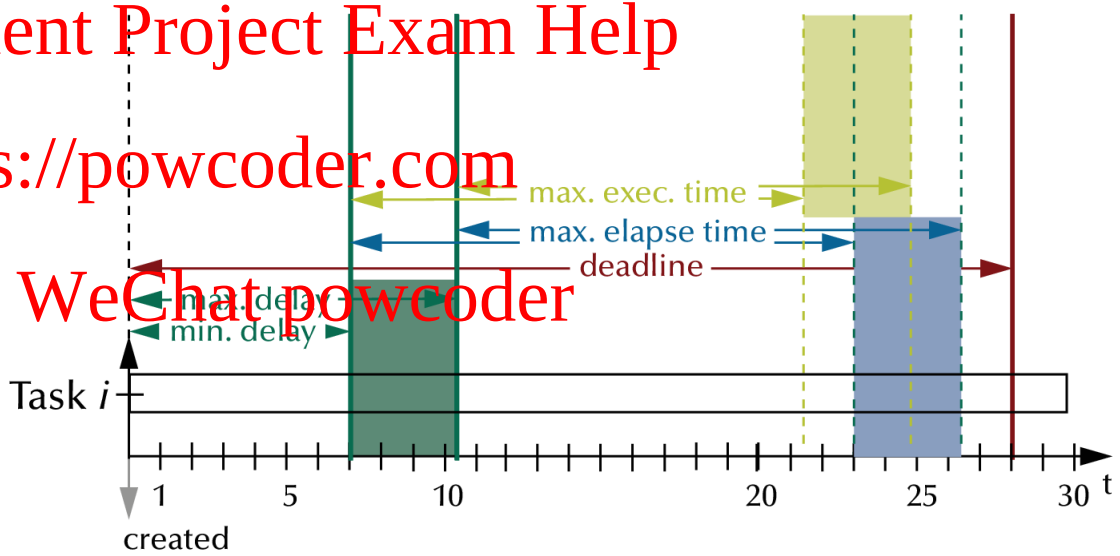
	Selection	Pre-emption	Waiting	Turnaround	Preferred Jobs	Starvation Possible?
Methods without priori knowledge of computation time C_i						
FCFS	max(W_i)	no	long	long avg, short max	equal	no
RR	equal share	yes	bounded	short avg, long max	short	no
MLFQ	priority queues	yes	very short	short avg, long max	short	no
Methods requiring knowledge of computation time C_i						
SJF	min(C_i)	no	medium	medium	short	yes
HRRF	$\max(\frac{W_i + C_i}{C_i})$	no	controllable	controllable	controllable	no
SRTF	min($C_i - E_i$)	yes	very short	variable	short	yes

Towards Predictable Scheduling

- Task requirements (quality of service):
  - Guarantee data flow levels
  - Guarantee reaction times
  - Guarantee deadlines
  - Guarantee delivery times
  - Provide bounds for the variations in results
- Examples:
  - Streaming media broadcasts, playing HD videos, live mixing audio/video
  - Reacting to users, Reacting to alarm situations
  - Delivering a signal to the physical world at the required time

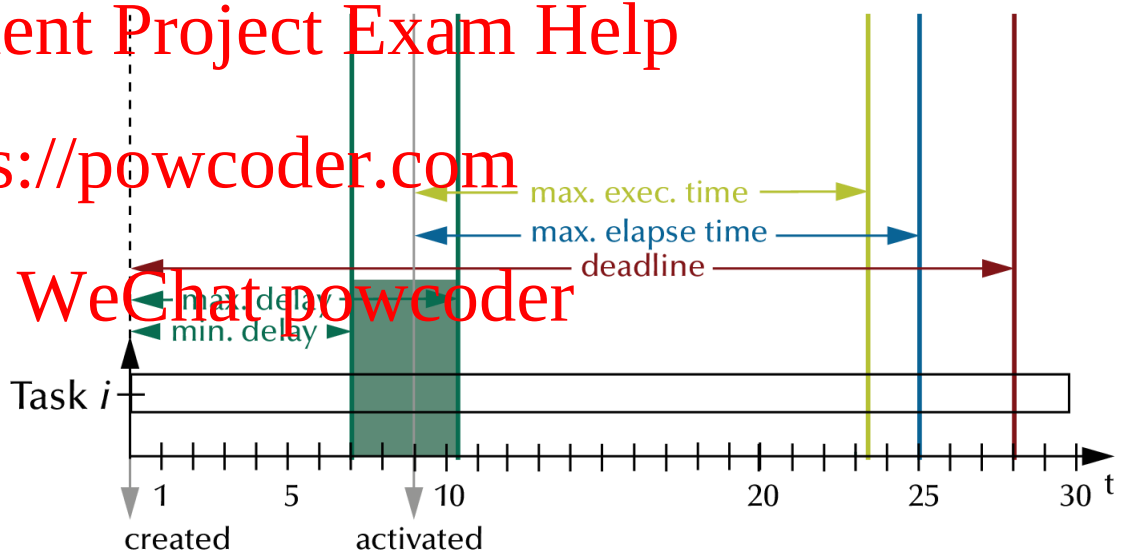
Temporal Scopes

- Common attributes:
  - Minimal & maximal delay after creation
  - Maximal elapsed time
  - Maximal execution time
  - Absolute deadline



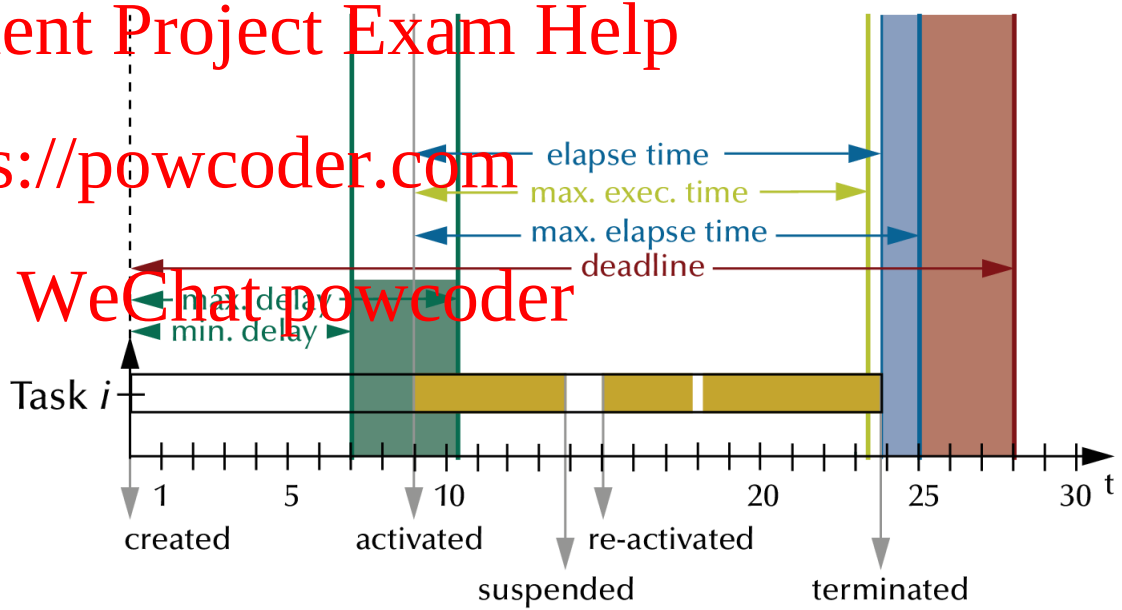
Temporal Scopes

- Common attributes:
  - Minimal & maximal delay after creation
  - Maximal elapsed time
  - Maximal execution time
  - Absolute deadline



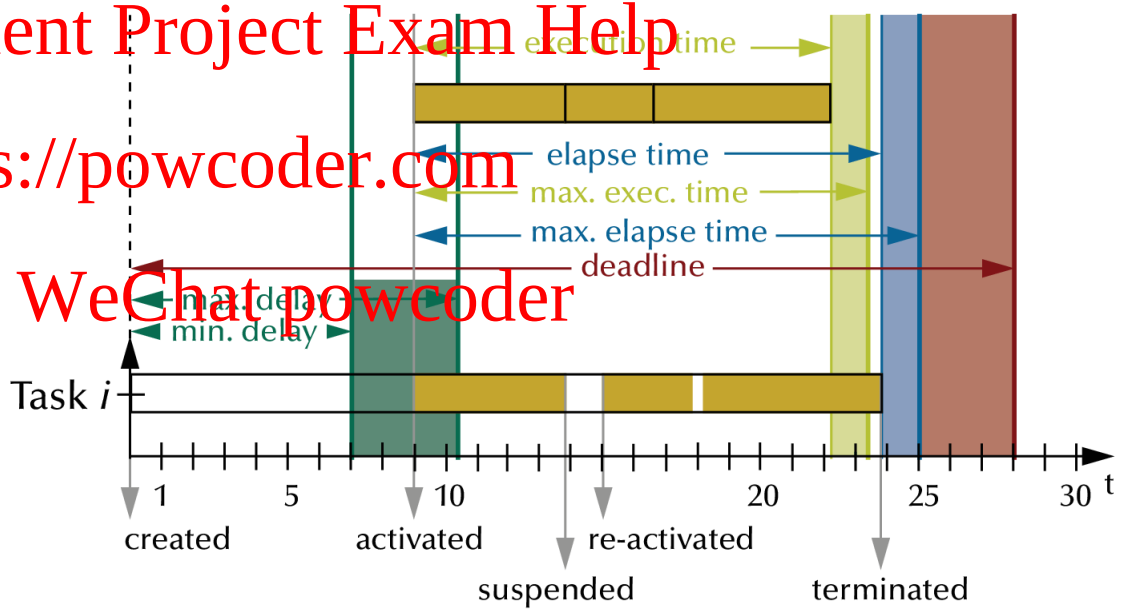
Temporal Scopes

- Common attributes:
  - Minimal & maximal delay after creation
  - Maximal elapsed time
  - Maximal execution time
  - Absolute deadline



Temporal Scopes

- Common attributes:
  - Minimal & maximal delay after creation
  - Maximal elapsed time
  - Maximal execution time
  - Absolute deadline



Common Temporal Scope Attributes

- Temporal scopes can be:
  - Periodic: controllers, routers, schedulers, streaming processes
  - Aperiodic: periodic ‘on average’ tasks, i.e. regular but not rigidly timed
  - Sporadic / Transient: user requests, alarms, I/O interaction
- Deadlines can be:
  - “Hard”: single failure leads to severe malfunction and/or disaster
  - “Firm”: results are meaningless after the deadline
    … only multiple or permanent failures lead to malfunction
  - “Soft”: results are still useful after the deadline