Response Time versus Utilization in Scheduler Overhead Accounting

Ana Sokolova
Department of Computer Sciences
University of Salzburg

joint work with
Silviu S. Craciunas and Christoph M. Kirsch

The problem

The problem

- scheduling processes in temporal isolation
  response time of a process (action) is independent of any concurrently running processes

The problem

- scheduling processes in temporal isolation
  response time of a process (action) is independent of any concurrently running processes
- response times include the scheduling overhead

The problem

doable with VBS

- scheduling processes in temporal isolation
  response time of a process (action) is independent of any concurrently running processes
- response times include the scheduling overhead

The problem

doable with VBS

- scheduling processes in temporal isolation
  response time of a process (action) is independent of any concurrently running processes
- response times include the scheduling overhead

accounting for it is the topic of this talk

The problem

doable with VBS

- scheduling processes in temporal isolation
  response time of a process (action) is independent of any concurrently running processes
- response times include the scheduling overhead

accounting for it is the topic of this talk

also doable with VBS

Process model

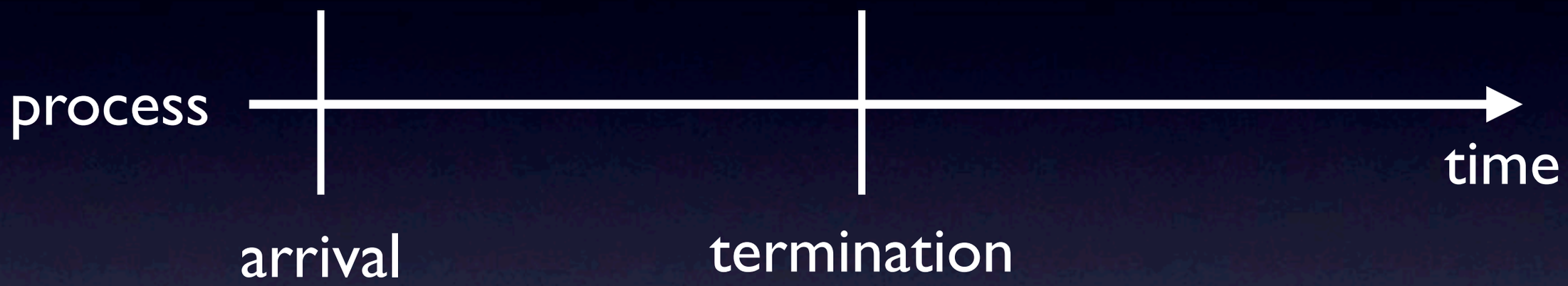
process

time

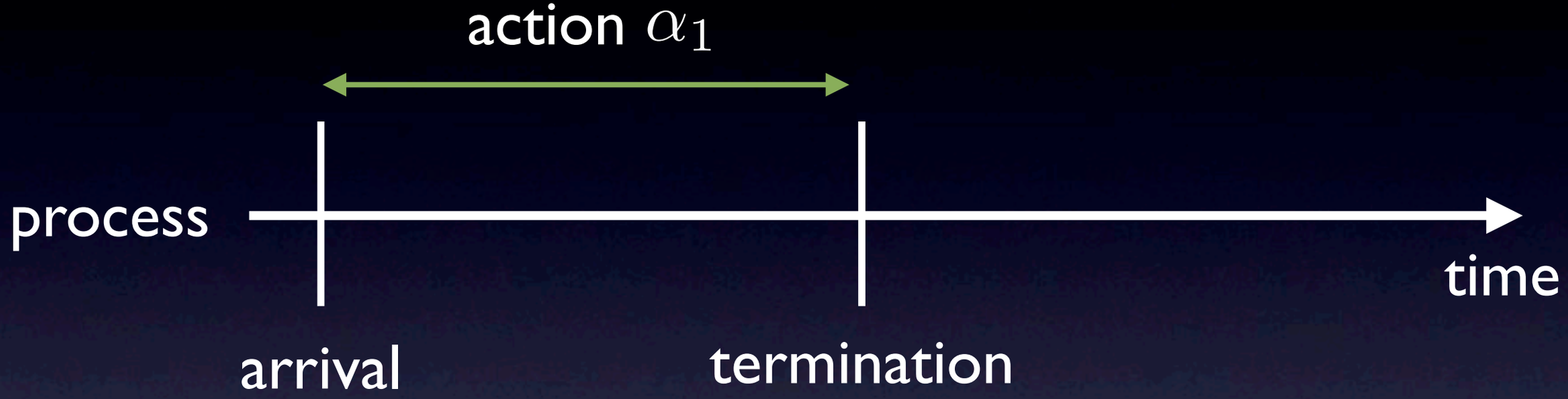
Process model

arrival

Process model

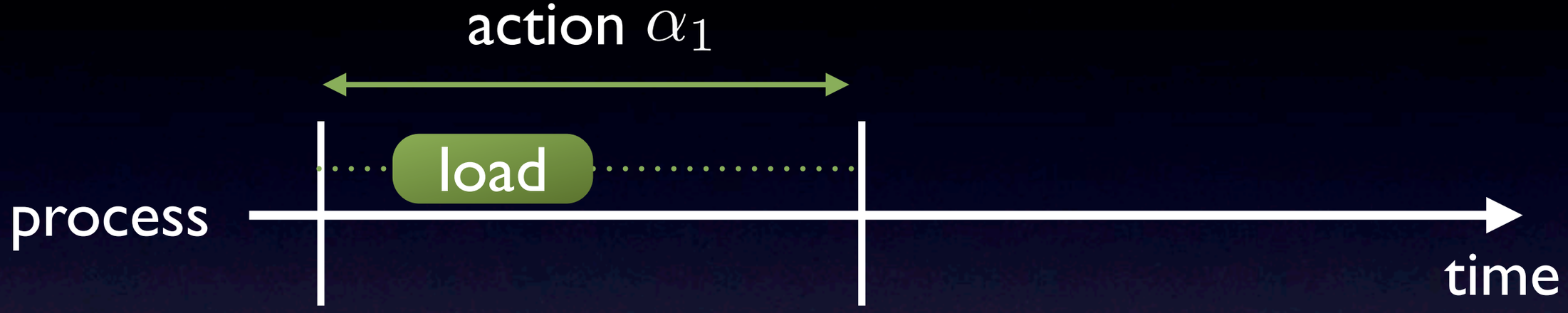


Process model



- action is a piece of code

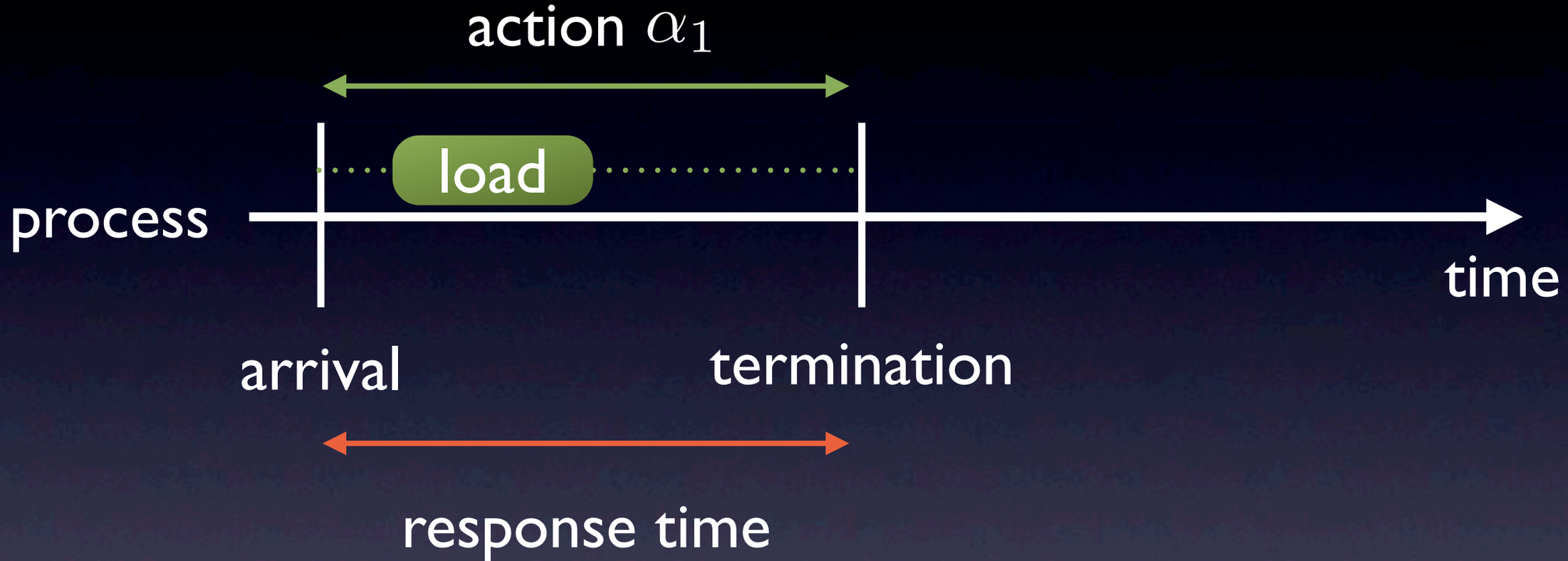
Process model



arrival termination

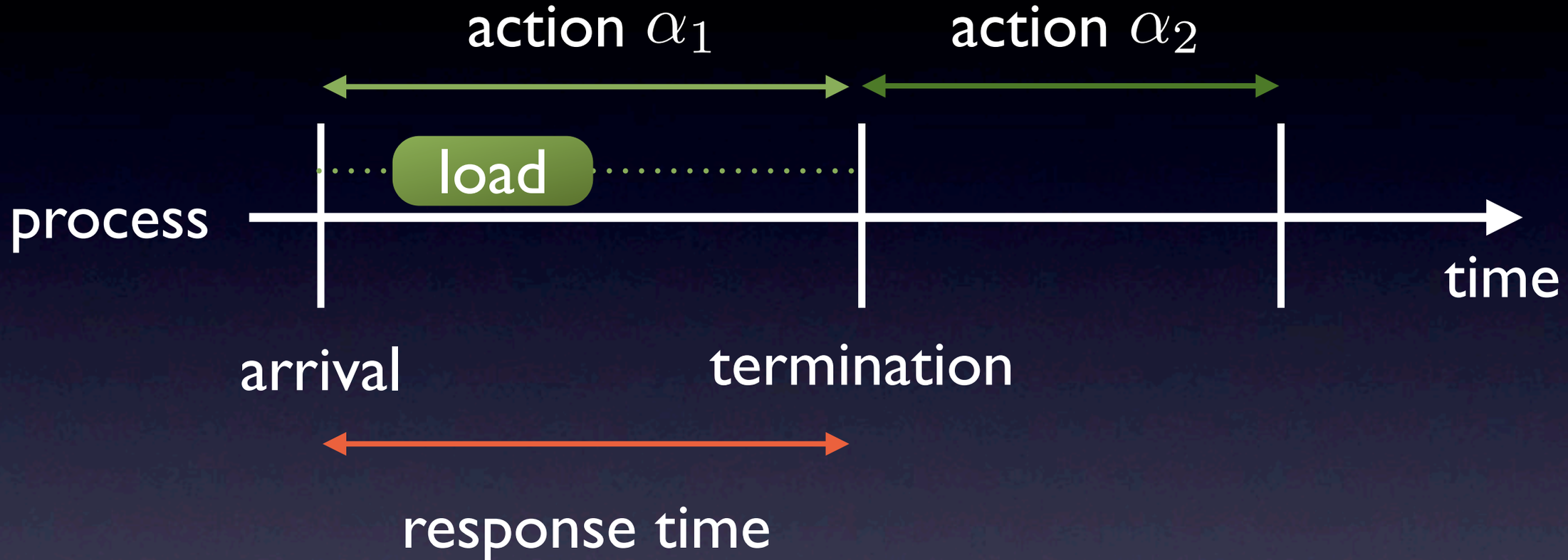
- action is a piece of code

Process model



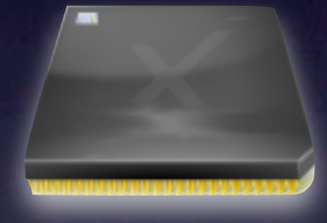
- action is a piece of code

Process model



- action is a piece of code
- process is a sequence of actions

Scheduling problem



schedule the processes so that each of their actions maintains its response time

Scheduling problem

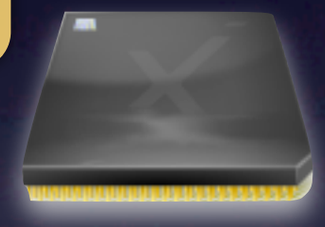
process 1

process 2

process n

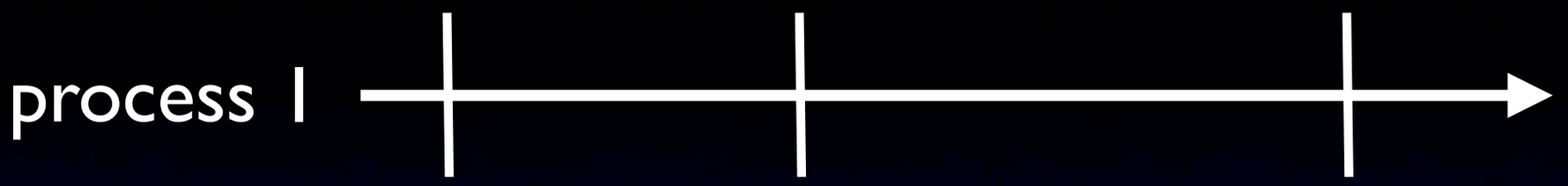
Solvable with variable-bandwidth servers (**VBS**)

uniprocessor



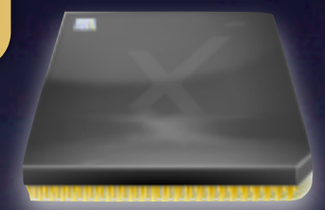
schedule the processes so that each of their actions maintains its response time

Scheduling problem



Solvable with variable-bandwidth servers (**VBS**)

uniprocessor



Results:

- constant-time scheduling algorithm
- constant time admission test

schedule the processes so that each of their actions maintains its response time

Resources and VBS

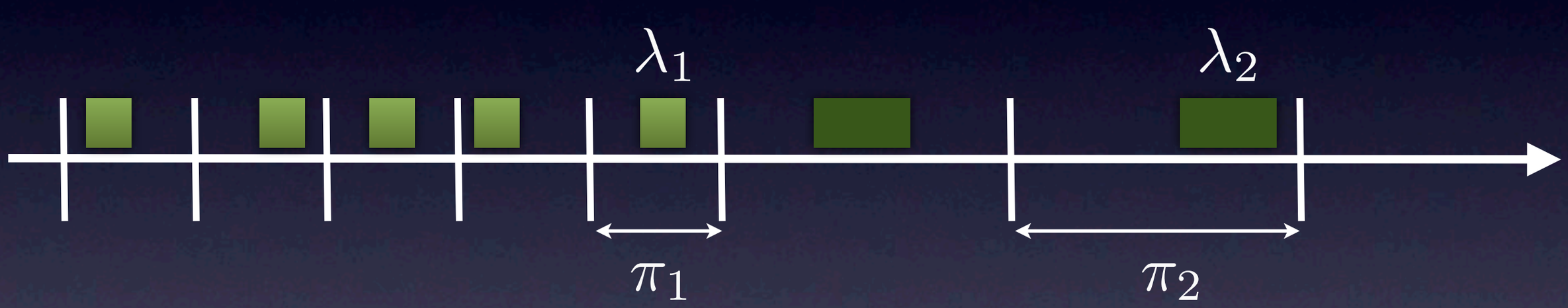
virtual periodic resources

period π limit λ utilization $\frac{\lambda}{\pi}$

Resources and VBS

virtual periodic resources

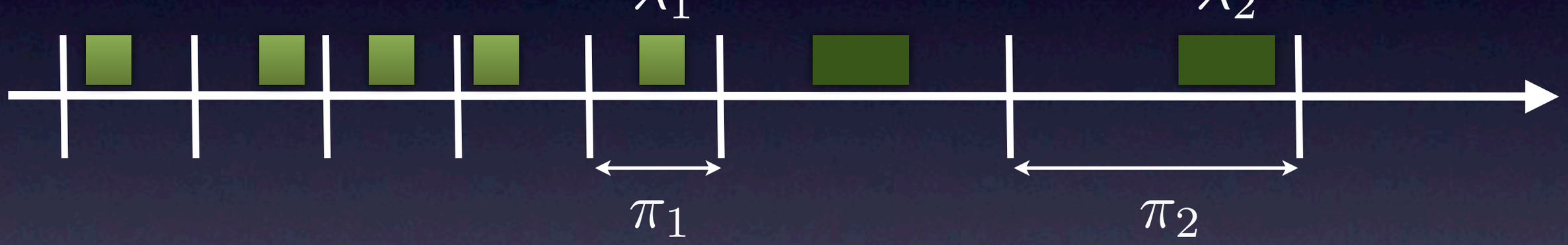
period π limit λ utilization $\frac{\lambda}{\pi}$



Resources and VBS

virtual periodic resources

period π limit λ utilization $\frac{\lambda}{\pi}$



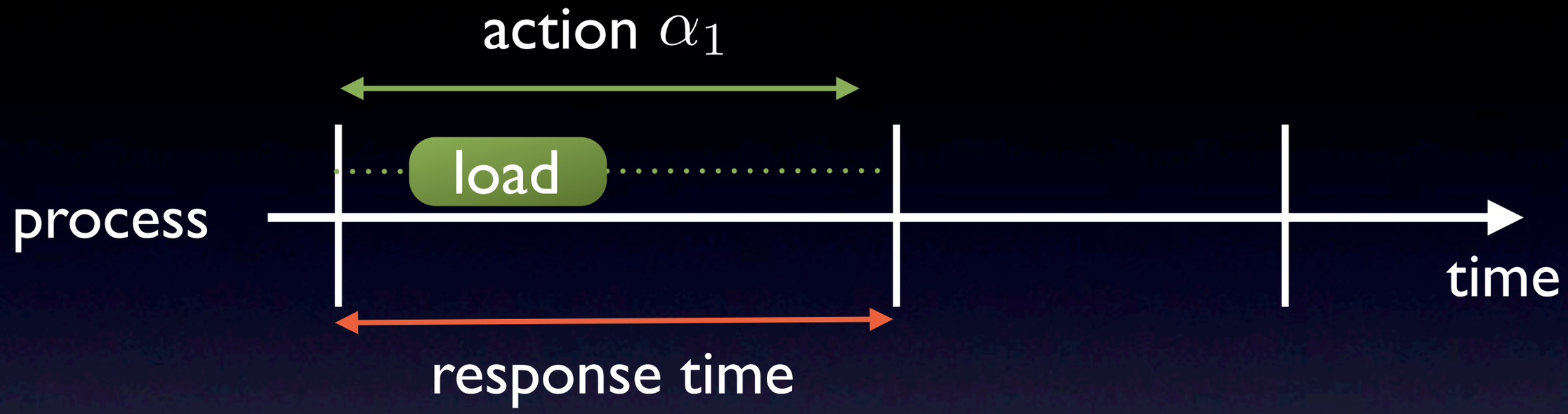
- VBS is determined by a bandwidth cap (u)
- VBS processes dynamically adjust speed (change resources)

$$\frac{\lambda_1}{\pi_1} \leq u \qquad \frac{\lambda_2}{\pi_2} \leq u$$

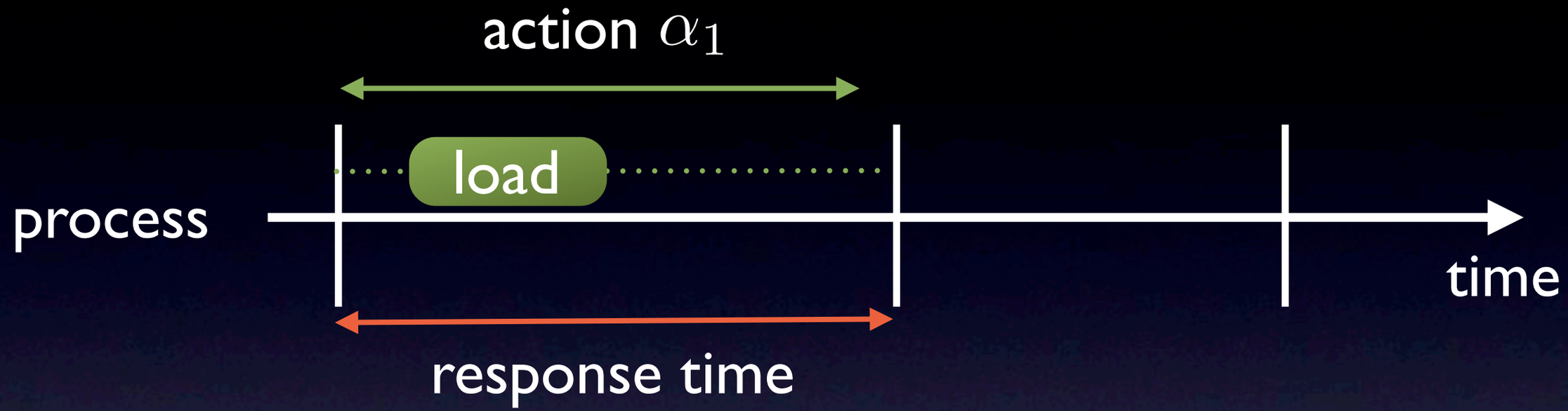
- generalization of constant bandwidth servers (CBS)

[Abeni and Buttazzo 2004]

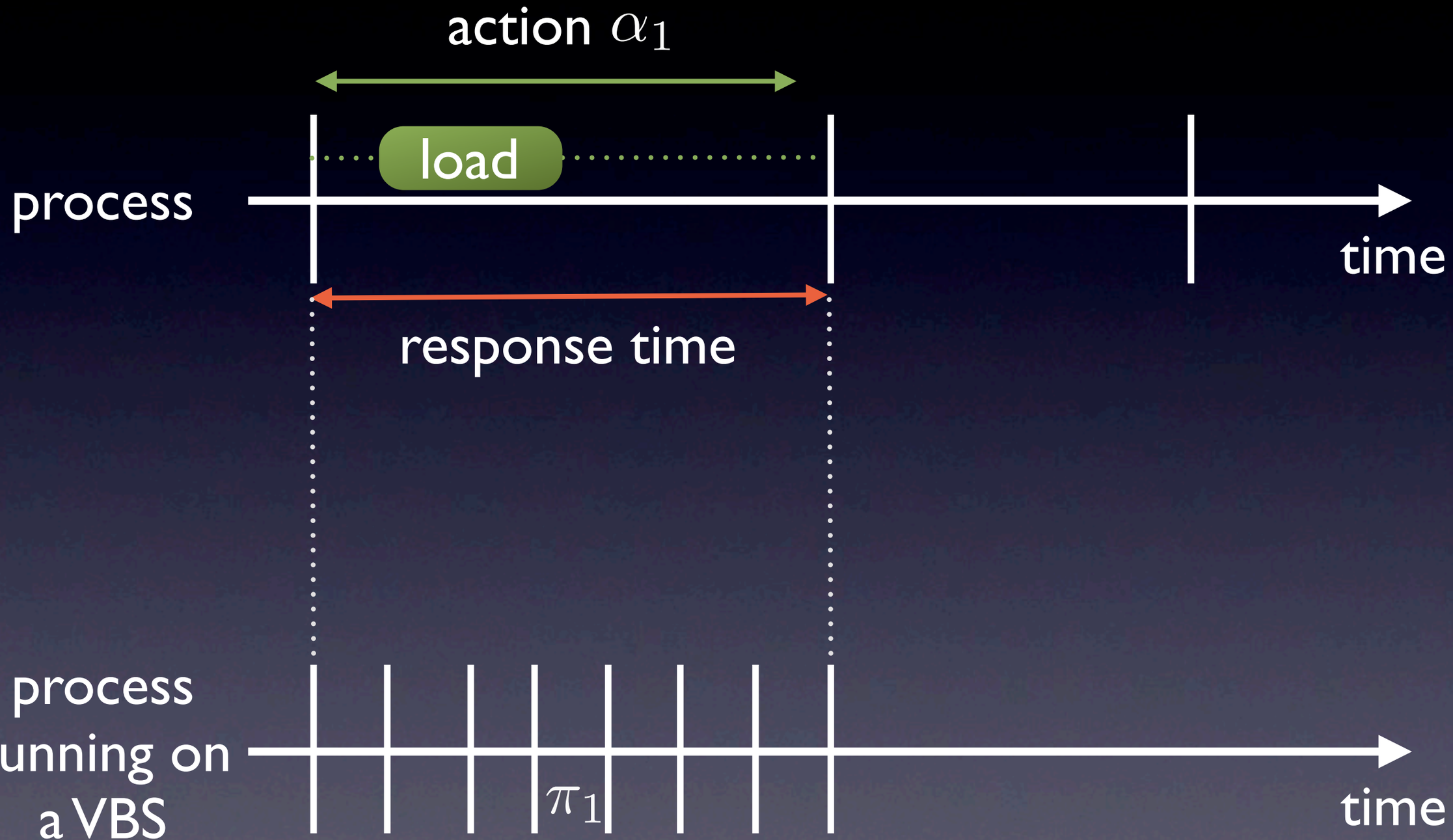
One process on a VBS



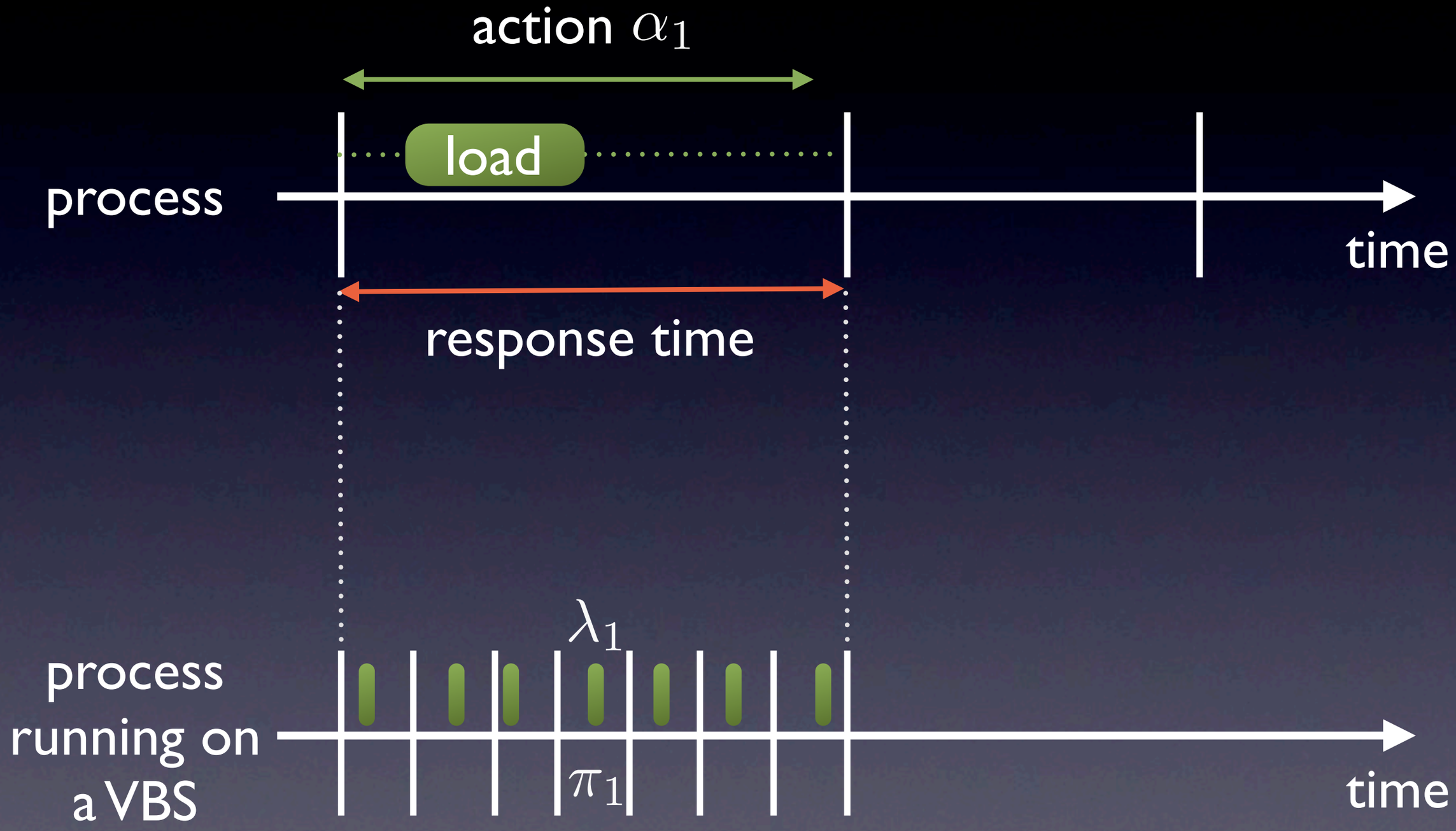
One process on a VBS



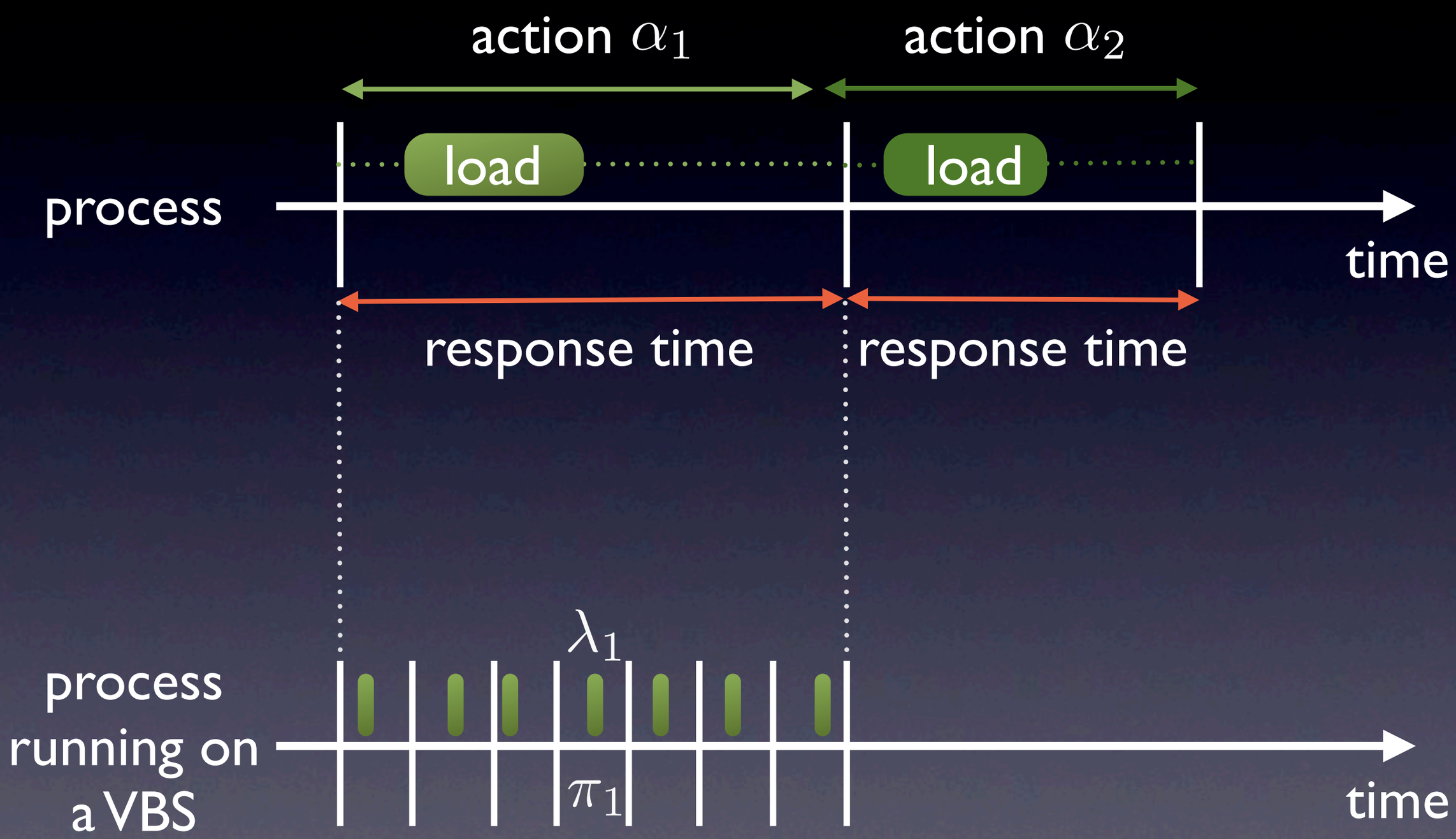
One process on a VBS



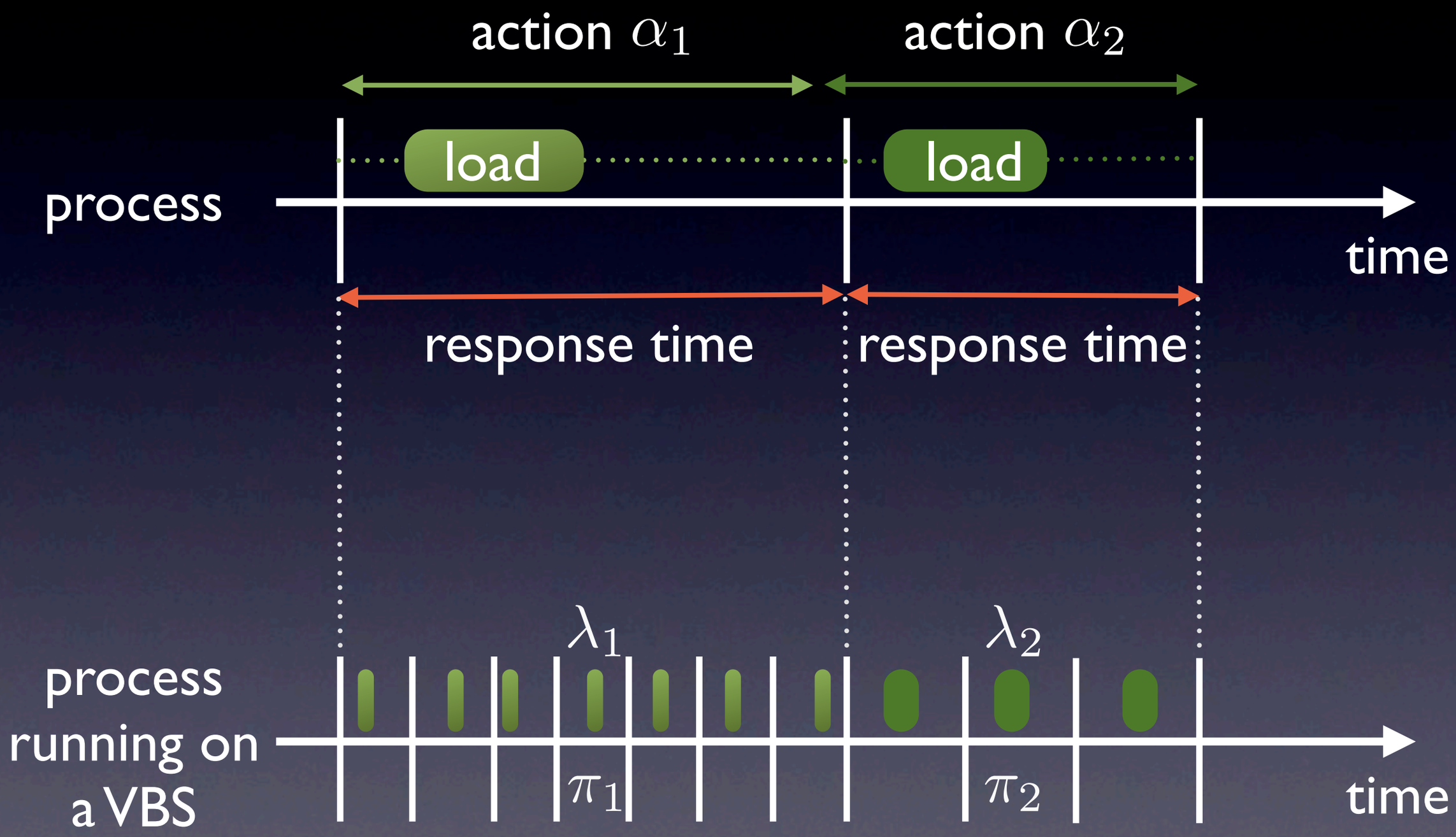
One process on a VBS



One process on a VBS



One process on a VBS



VBS

VBS

VBS

VBS

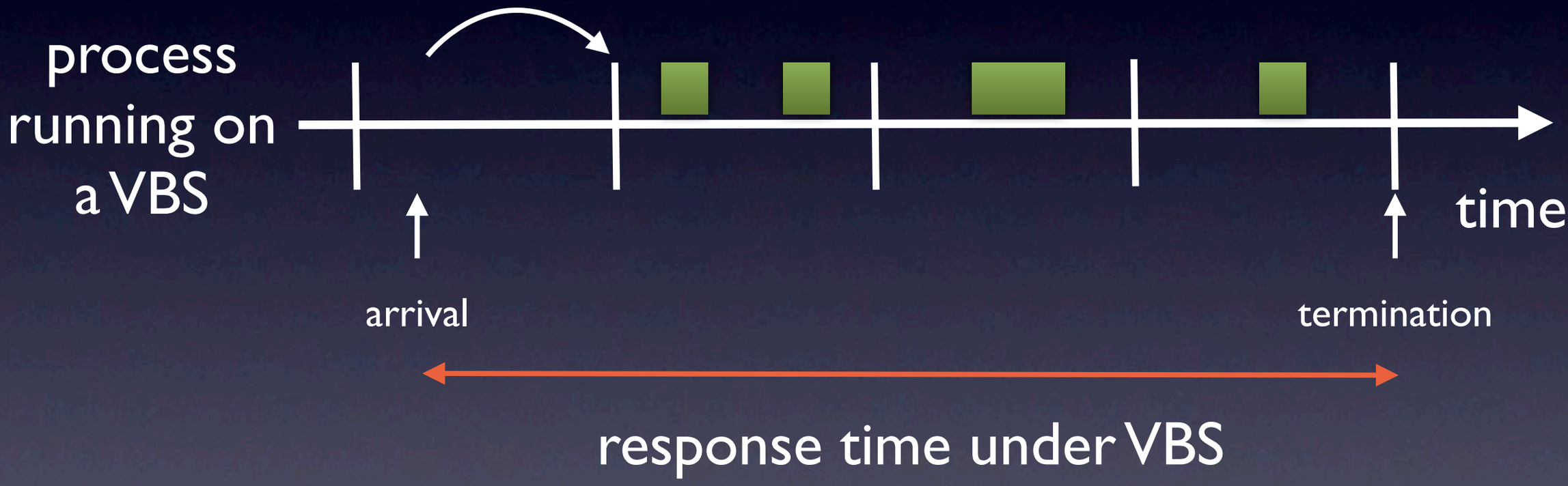
VBS

VBS

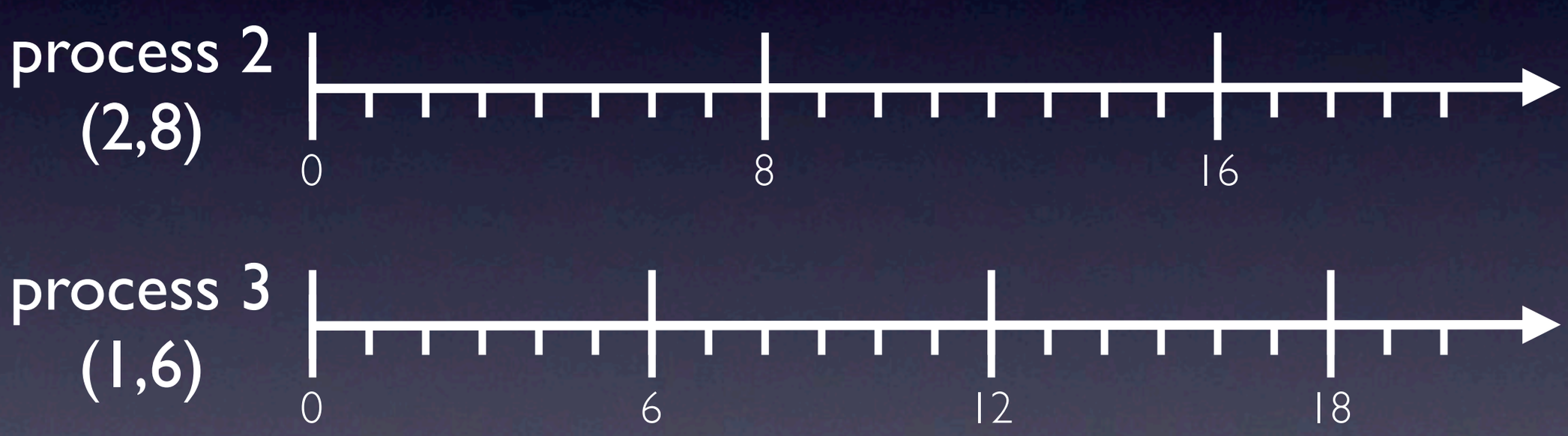
VBS

VBS

VBS

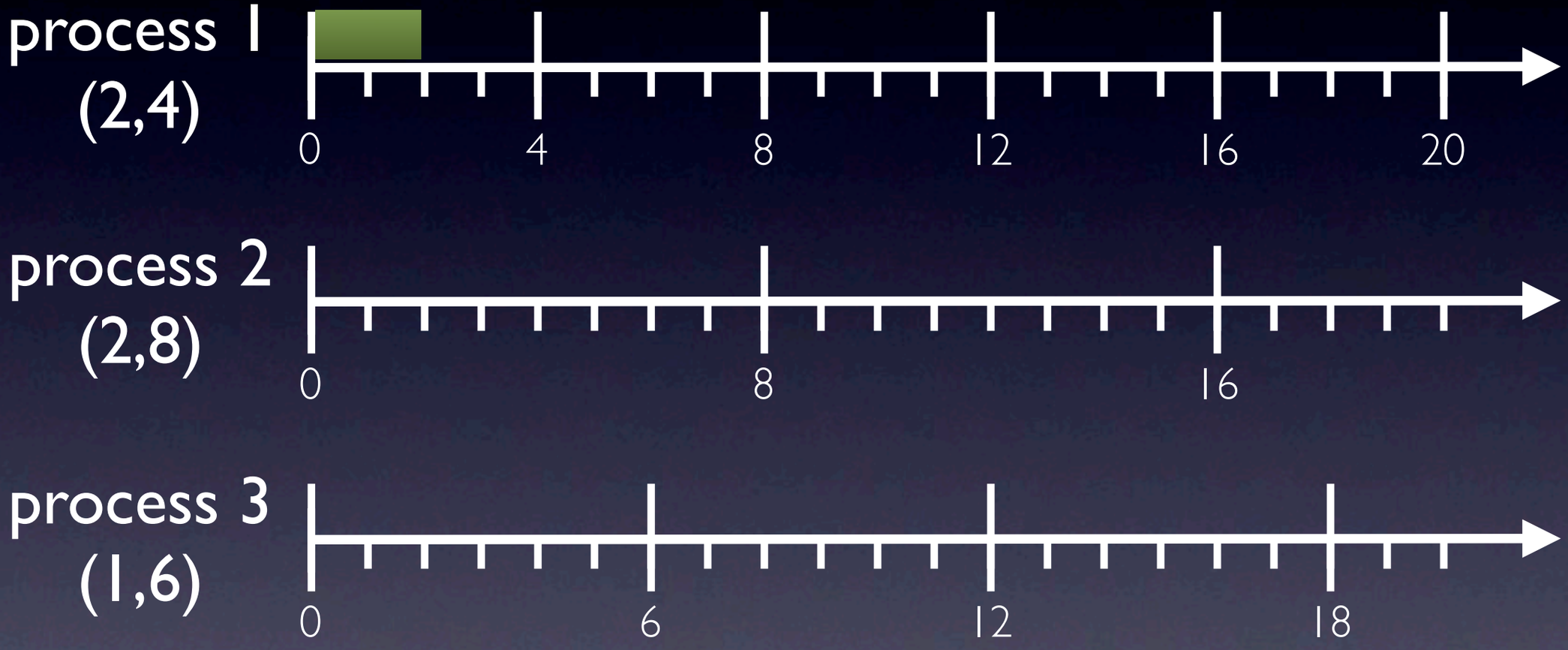

process
running on
a VBS
time
arrival
termination
response time under VBS

VBS



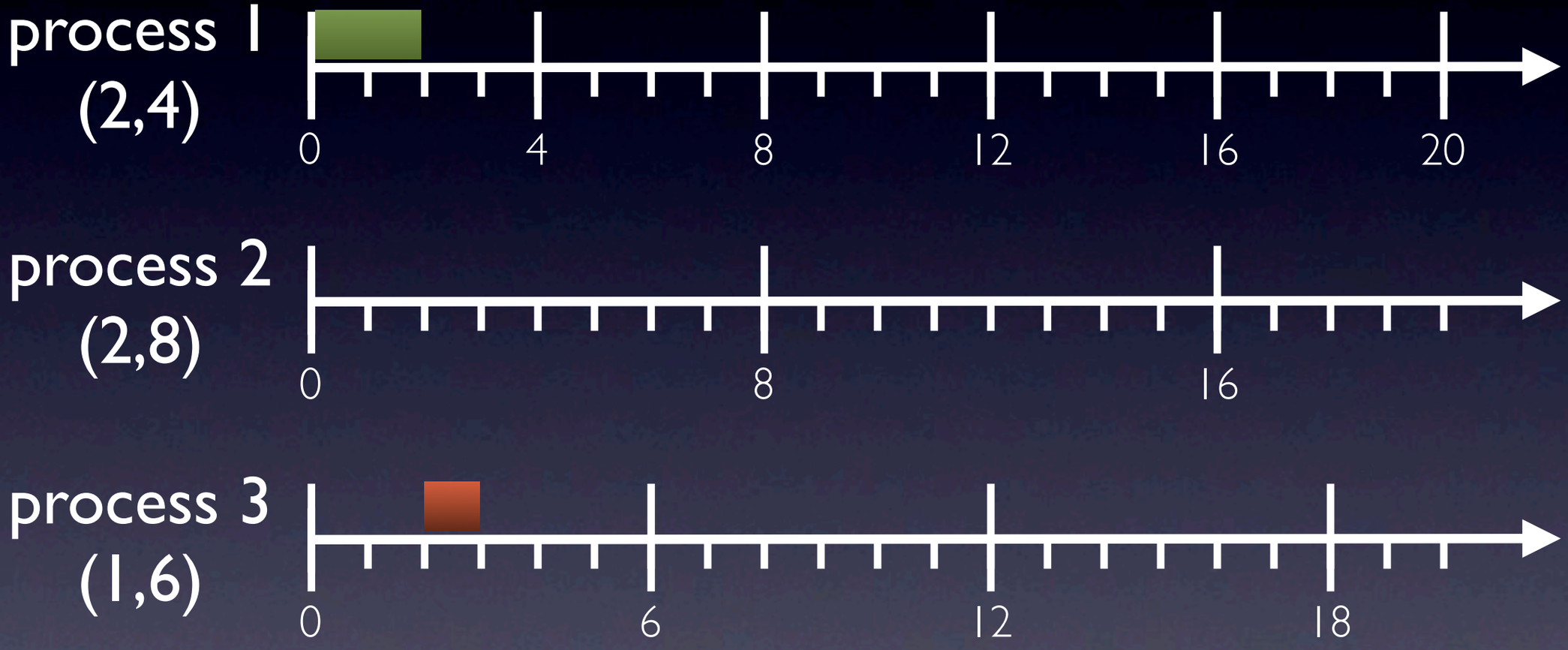
multiple processes are EDF-scheduled

VBS



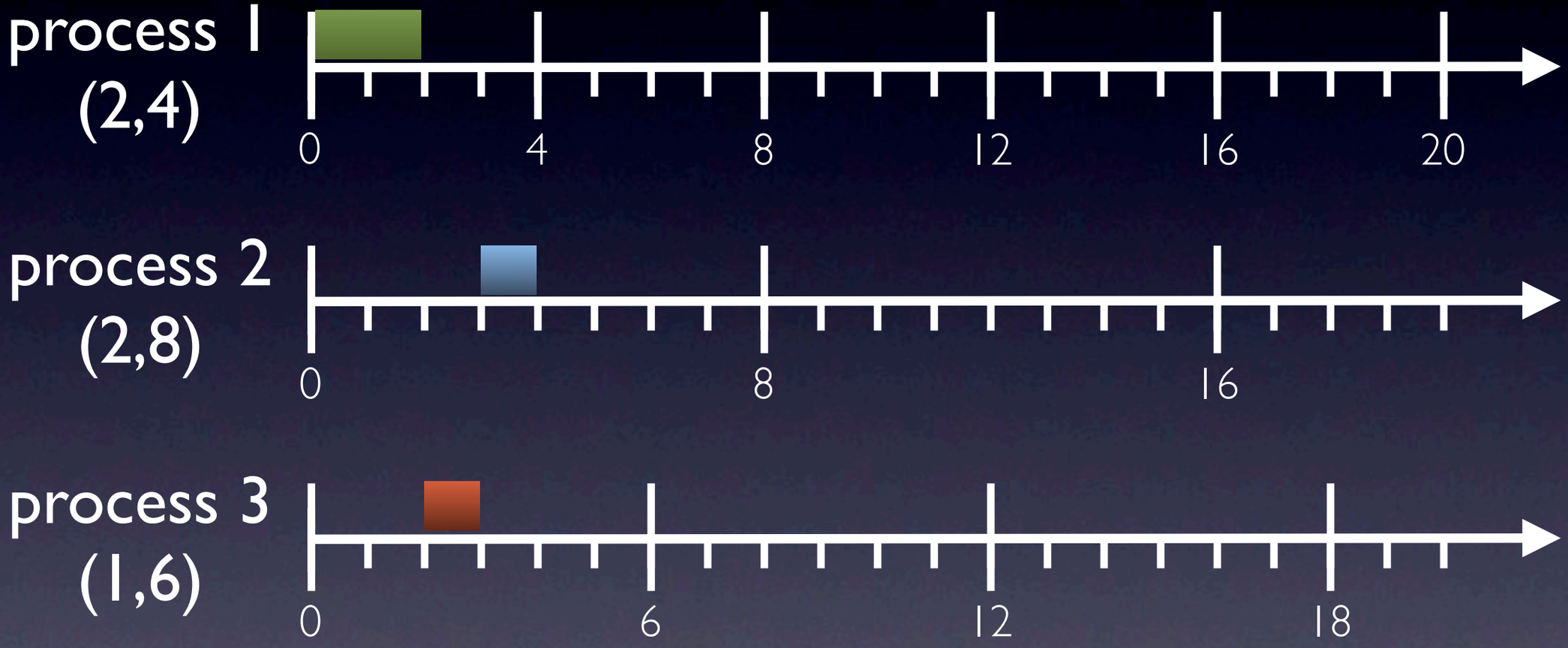
multiple processes are EDF-scheduled

VBS



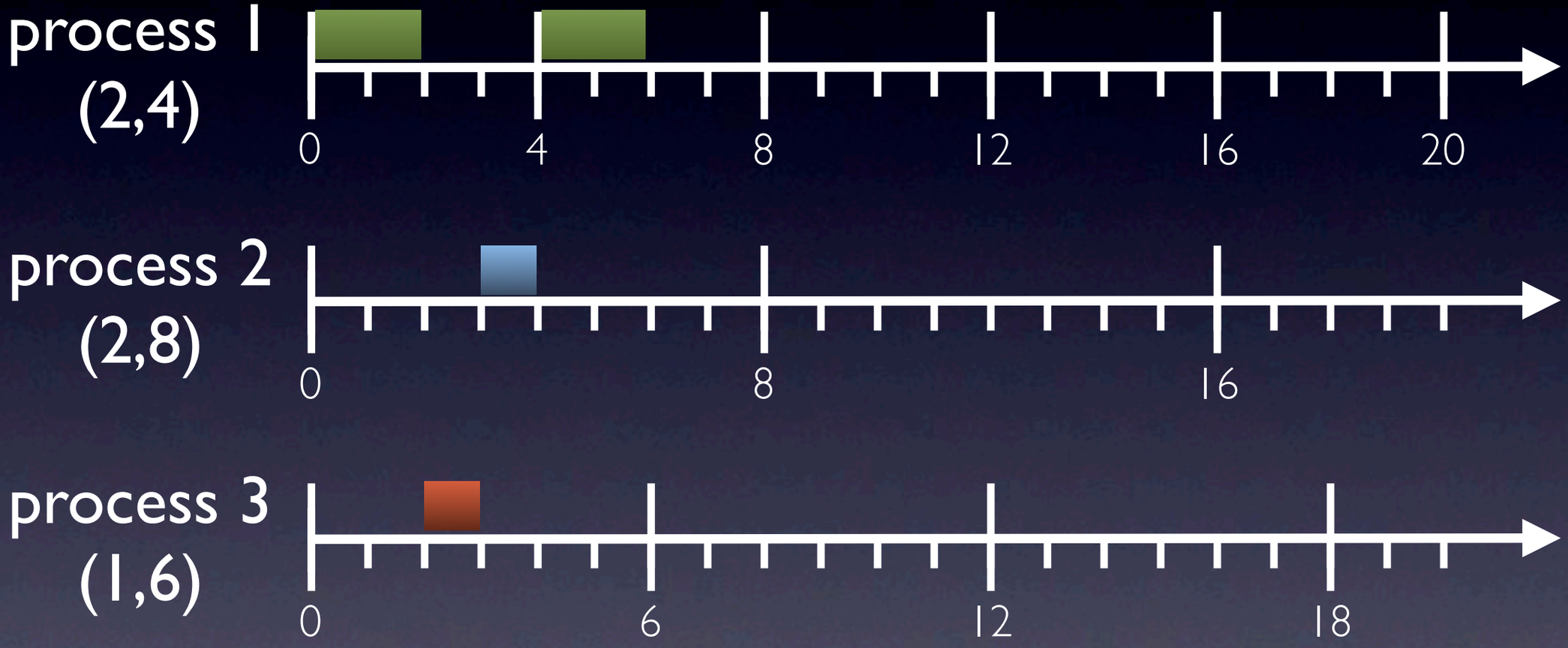
multiple processes are EDF-scheduled

VBS



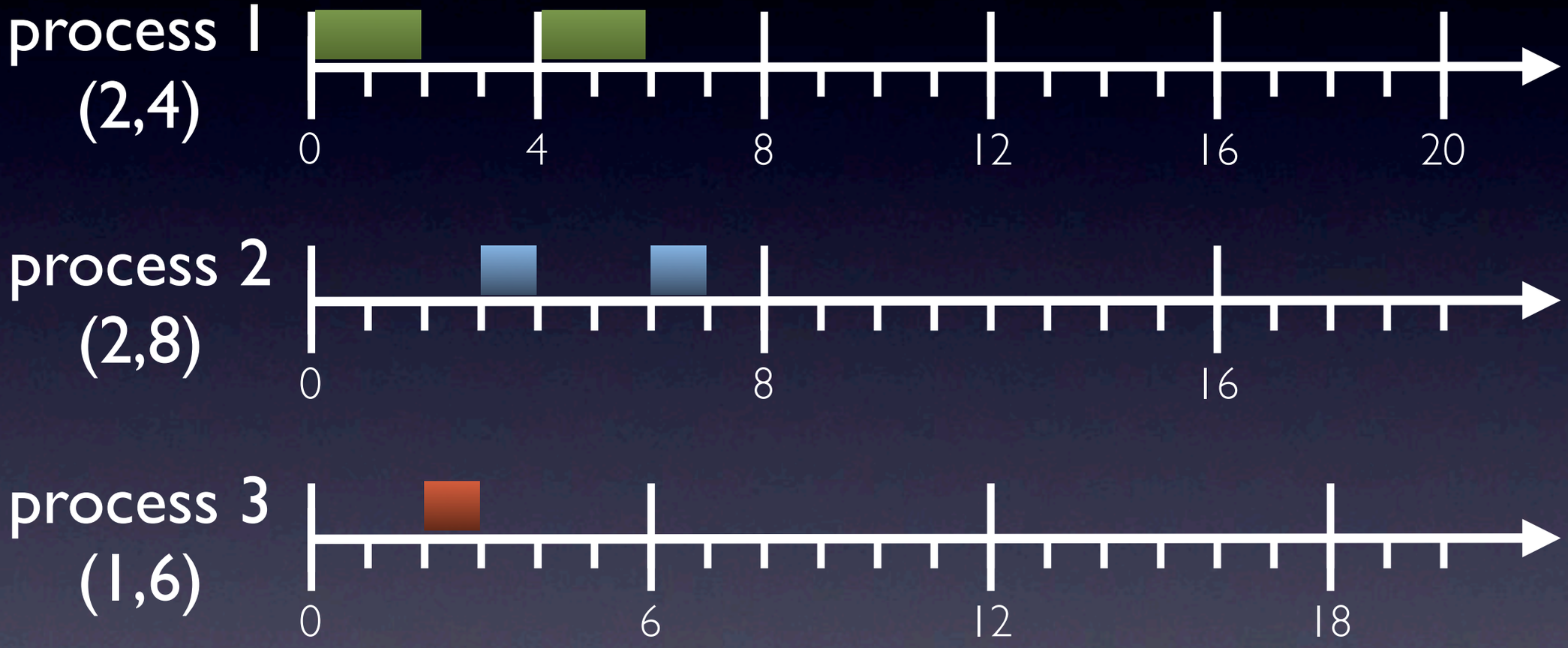
multiple processes are EDF-scheduled

VBS



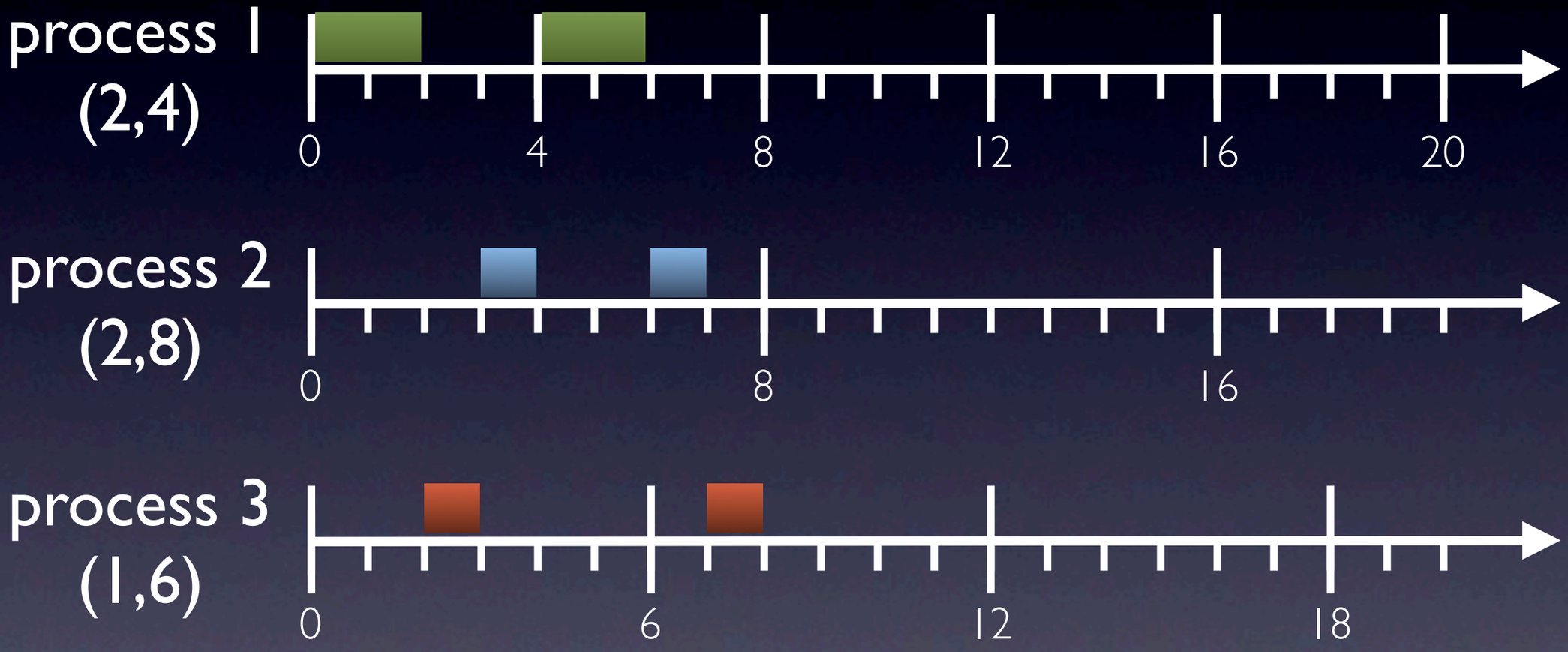
multiple processes are EDF-scheduled

VBS



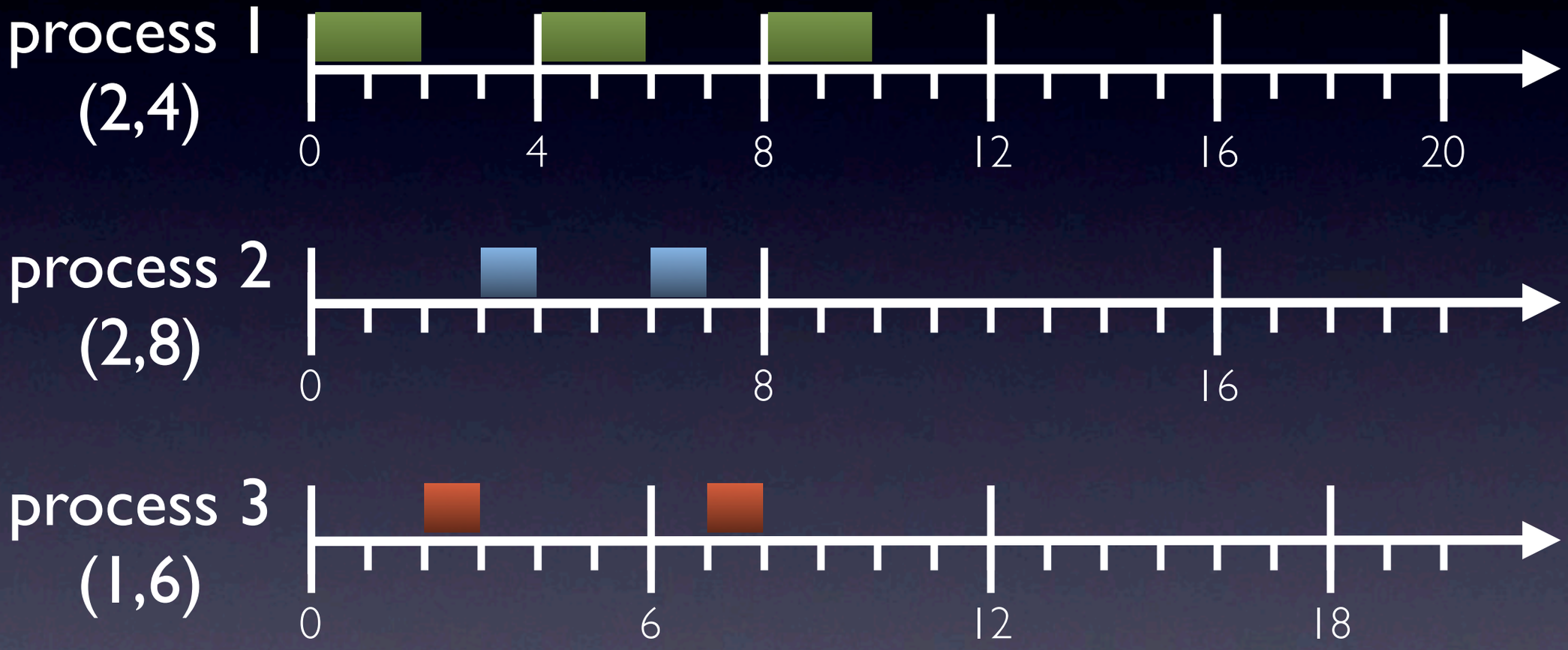
multiple processes are EDF-scheduled

VBS



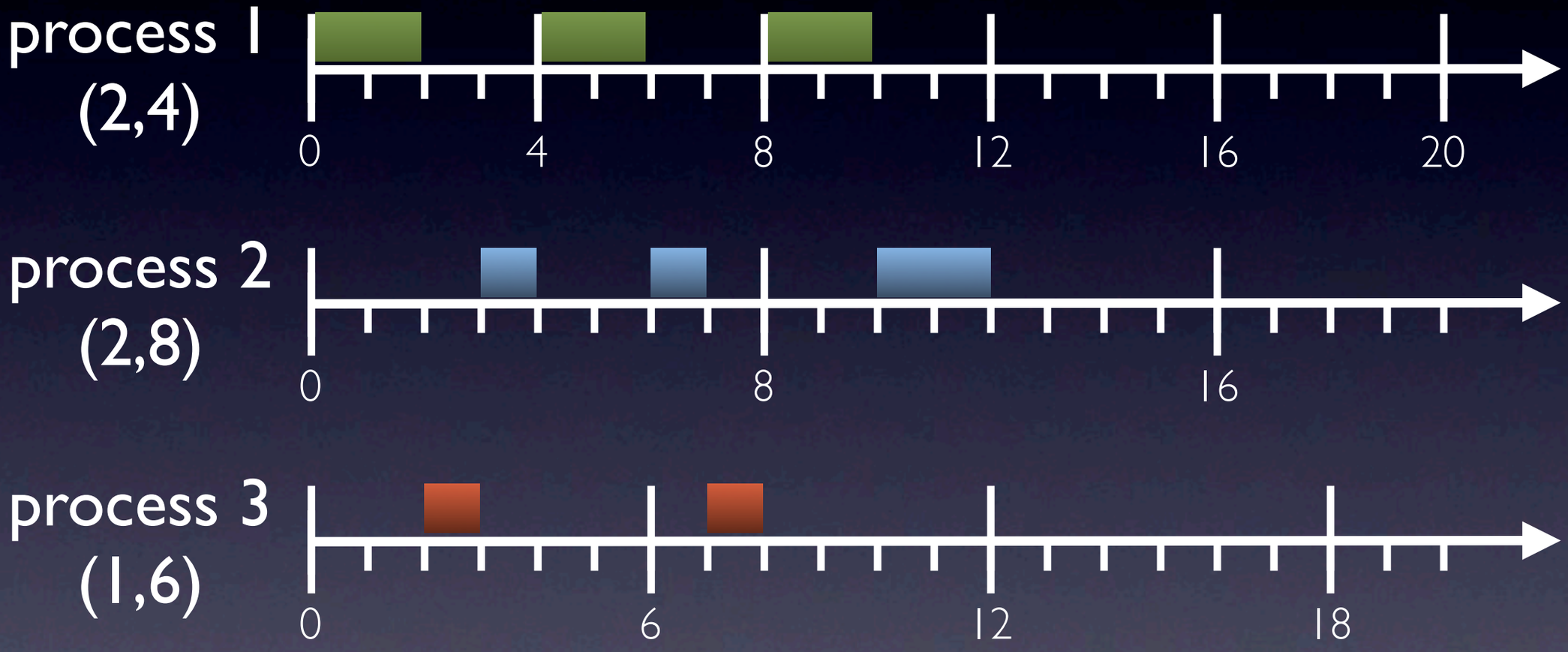
multiple processes are EDF-scheduled

VBS



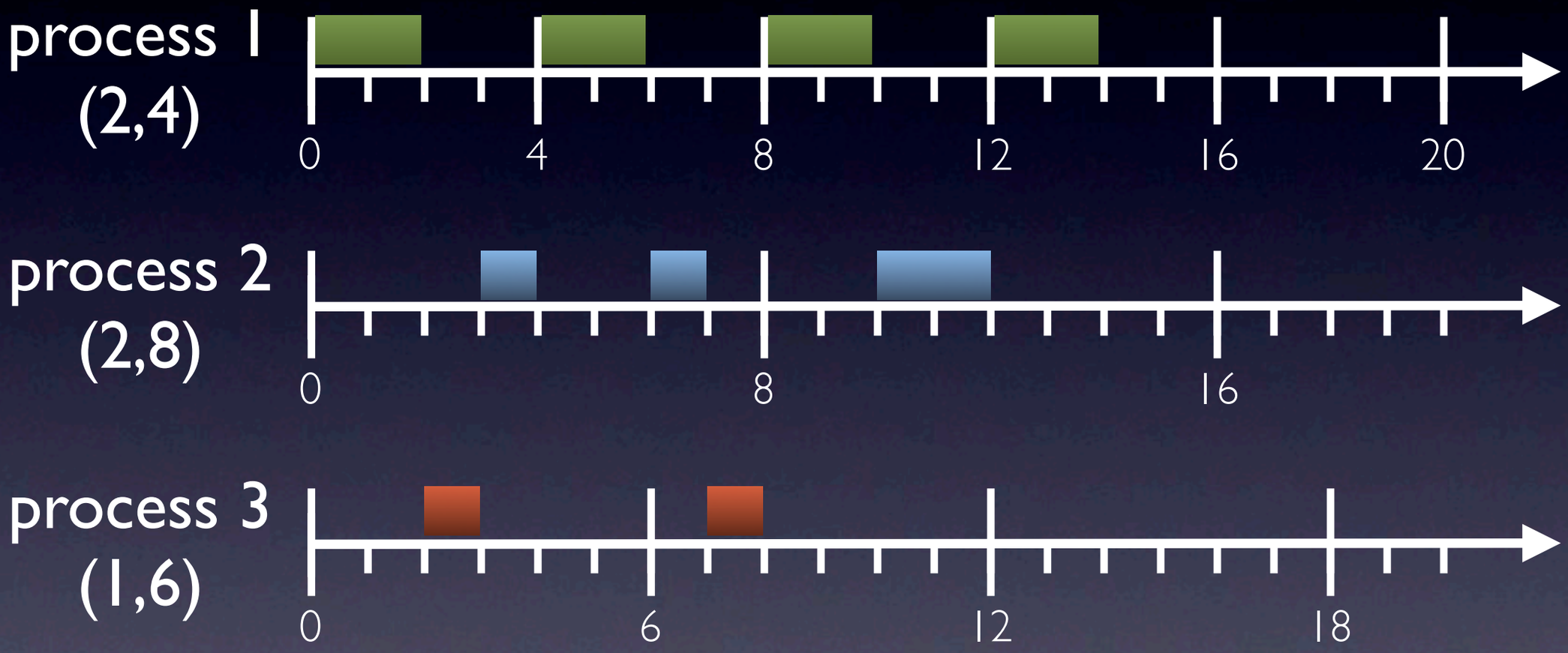
multiple processes are EDF-scheduled

VBS



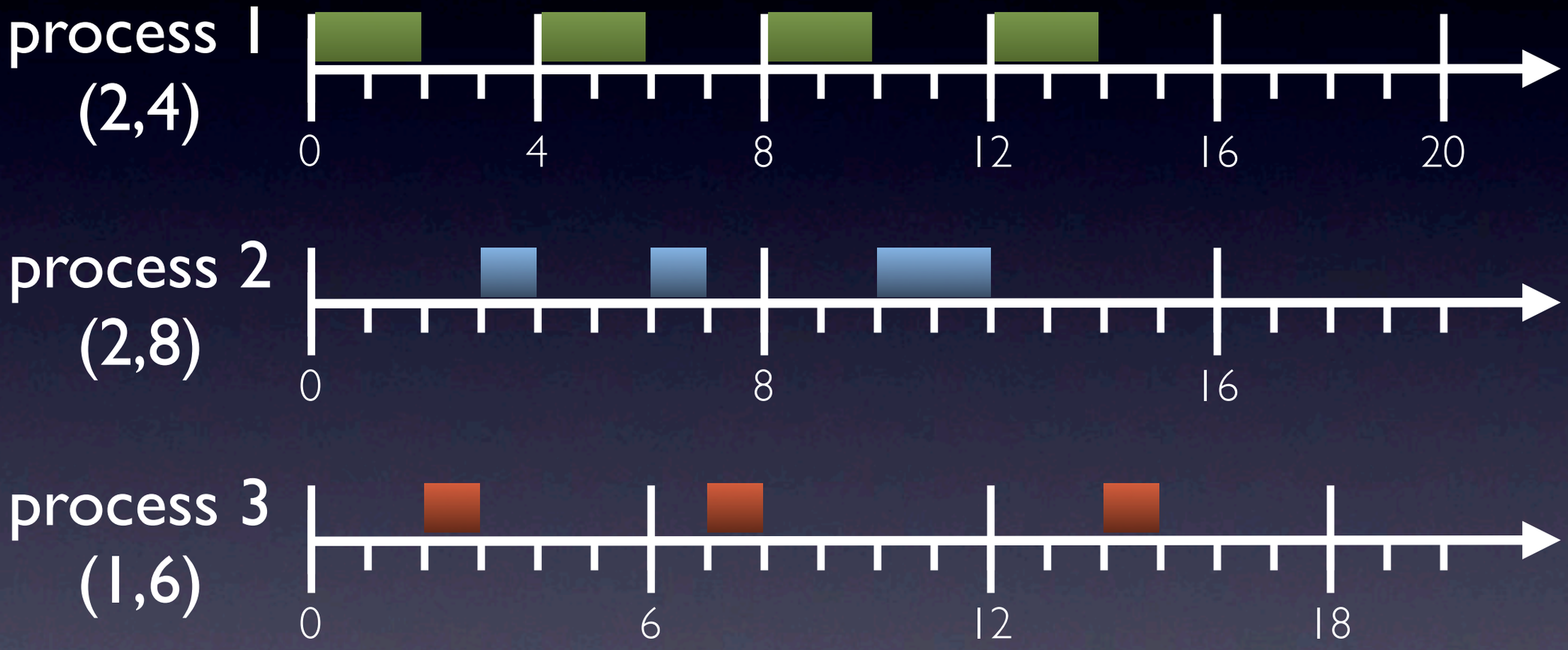
multiple processes are EDF-scheduled

VBS



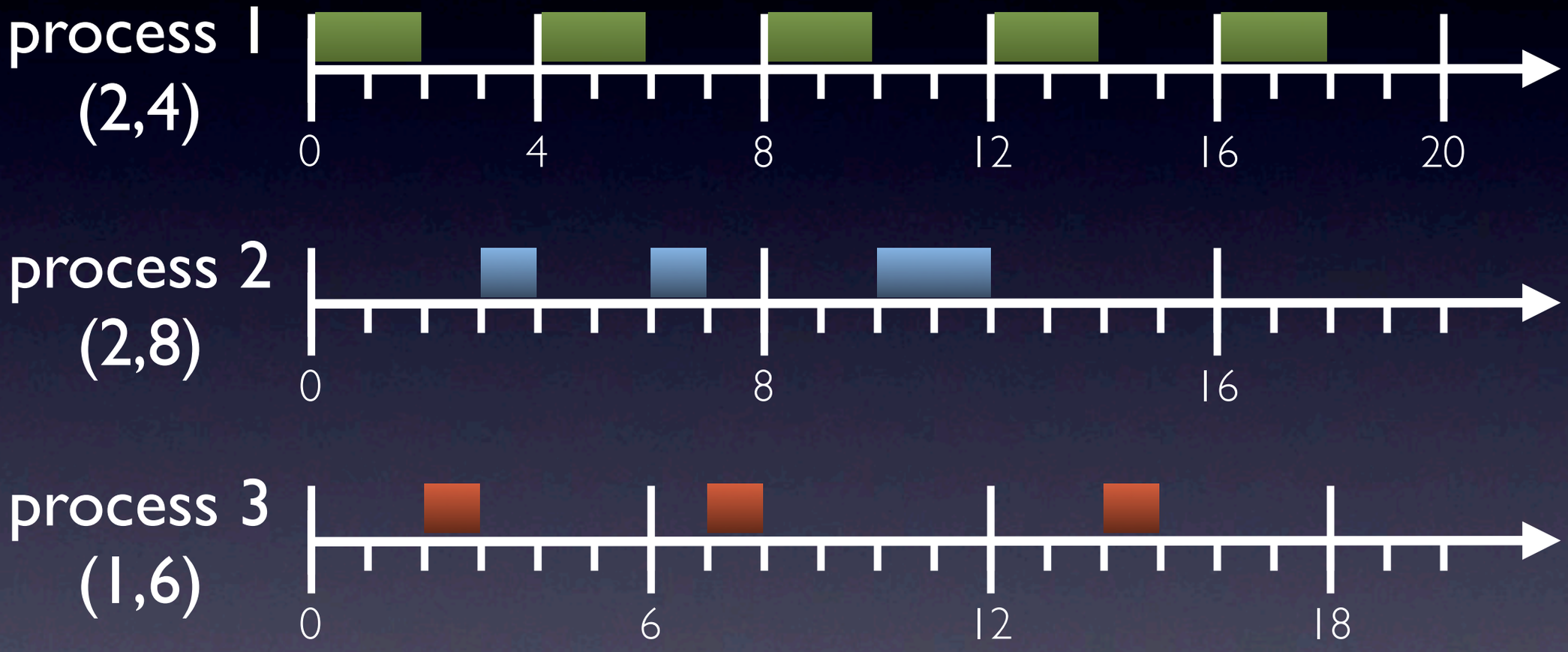
multiple processes are EDF-scheduled

VBS



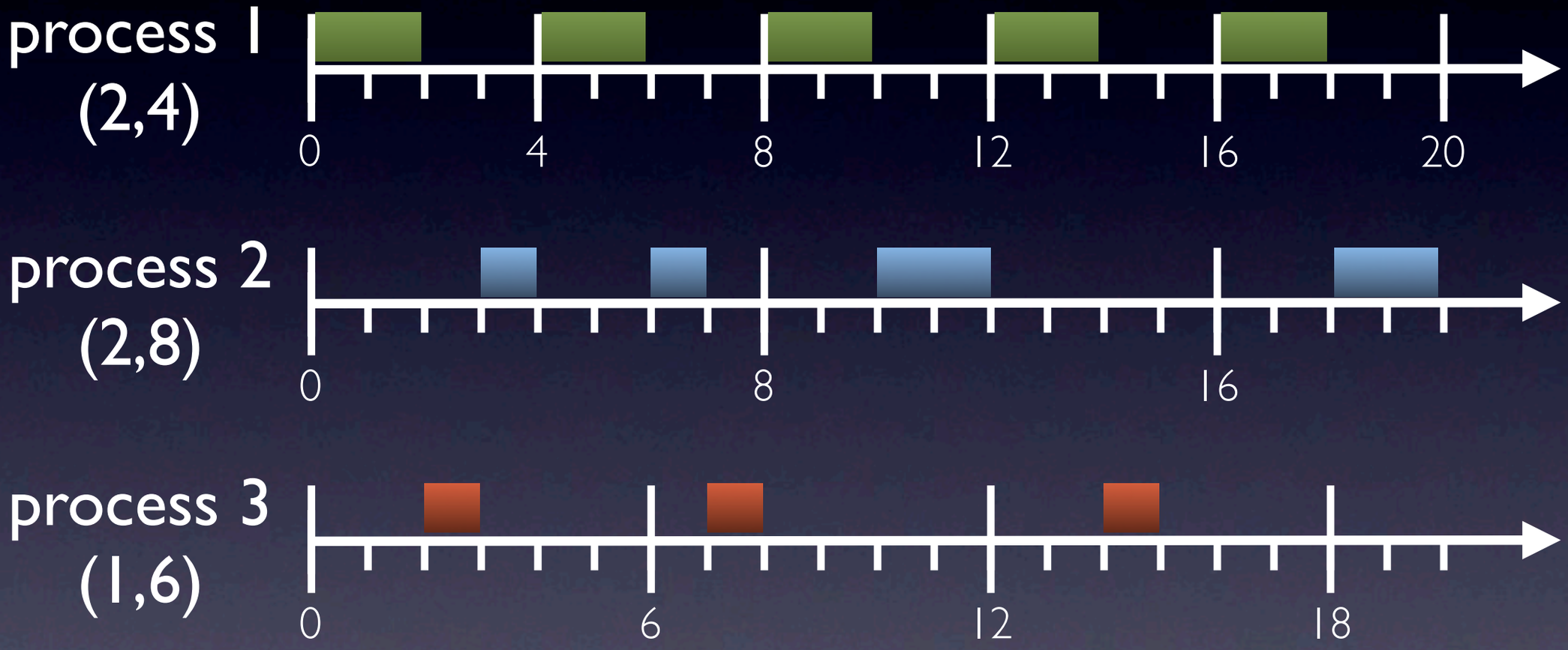
multiple processes are EDF-scheduled

VBS



multiple processes are EDF-scheduled

VBS



multiple processes are EDF-scheduled

Scheduling result and bounds

Processes $P_1, P_2, \ldots, P_n$ on VBSs $u_1, u_2, \ldots, u_n$ are schedulable if $\sum_{i=1}^{n} u_i \leq 1$

Scheduling result and bounds

a sufficient condition

Processes $P_1, P_2, \ldots, P_n$ on VBSs $u_1, u_2, \ldots, u_n$ are schedulable if $\sum_{i=1}^{n} u_i \leq 1$

Scheduling result and bounds

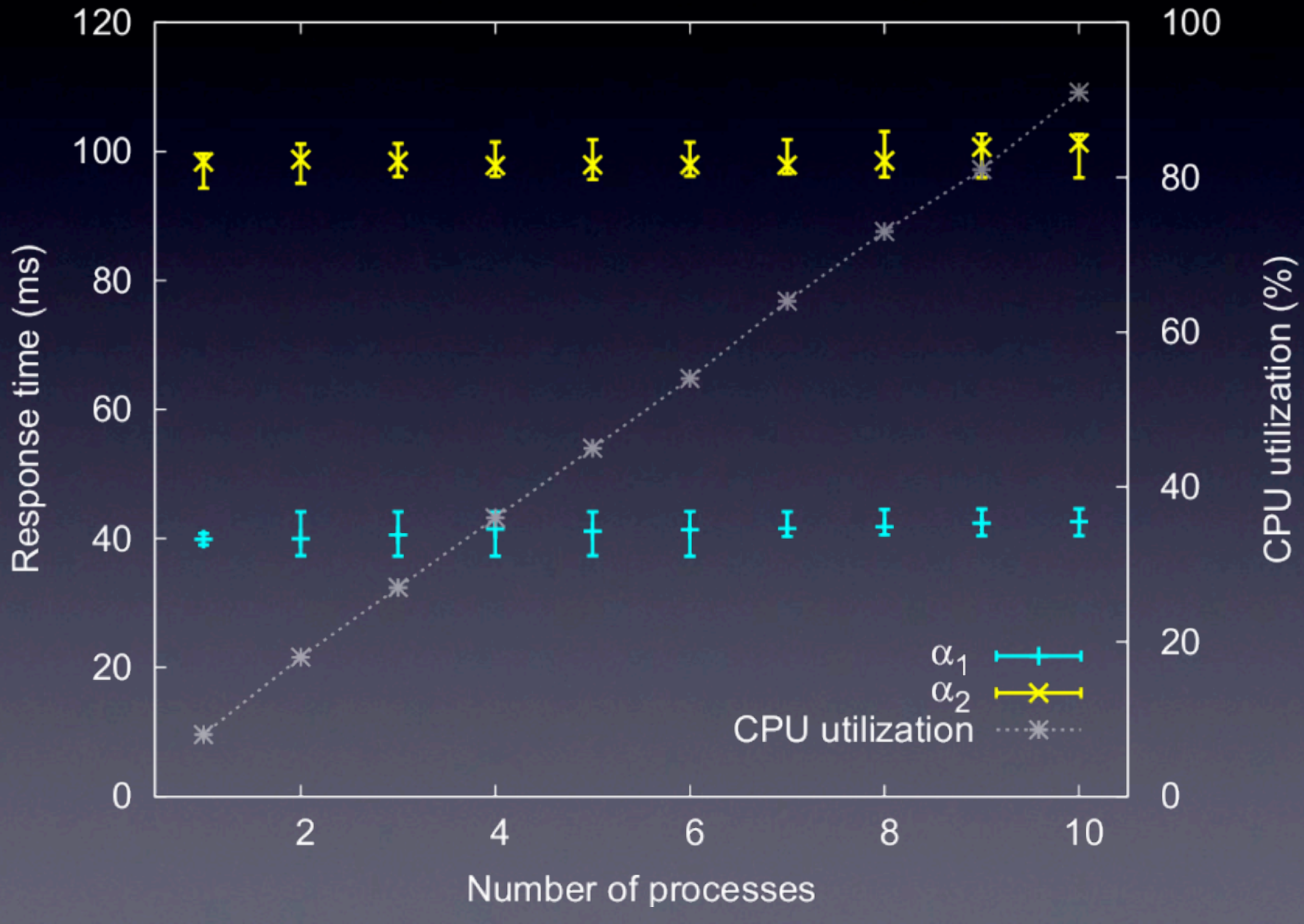
a sufficient condition

Processes $P_1, P_2, \ldots, P_n$ on VBSs $u_1, u_2, \ldots, u_n$ are schedulable if $\sum_{i=1}^{n} u_i \leq 1$

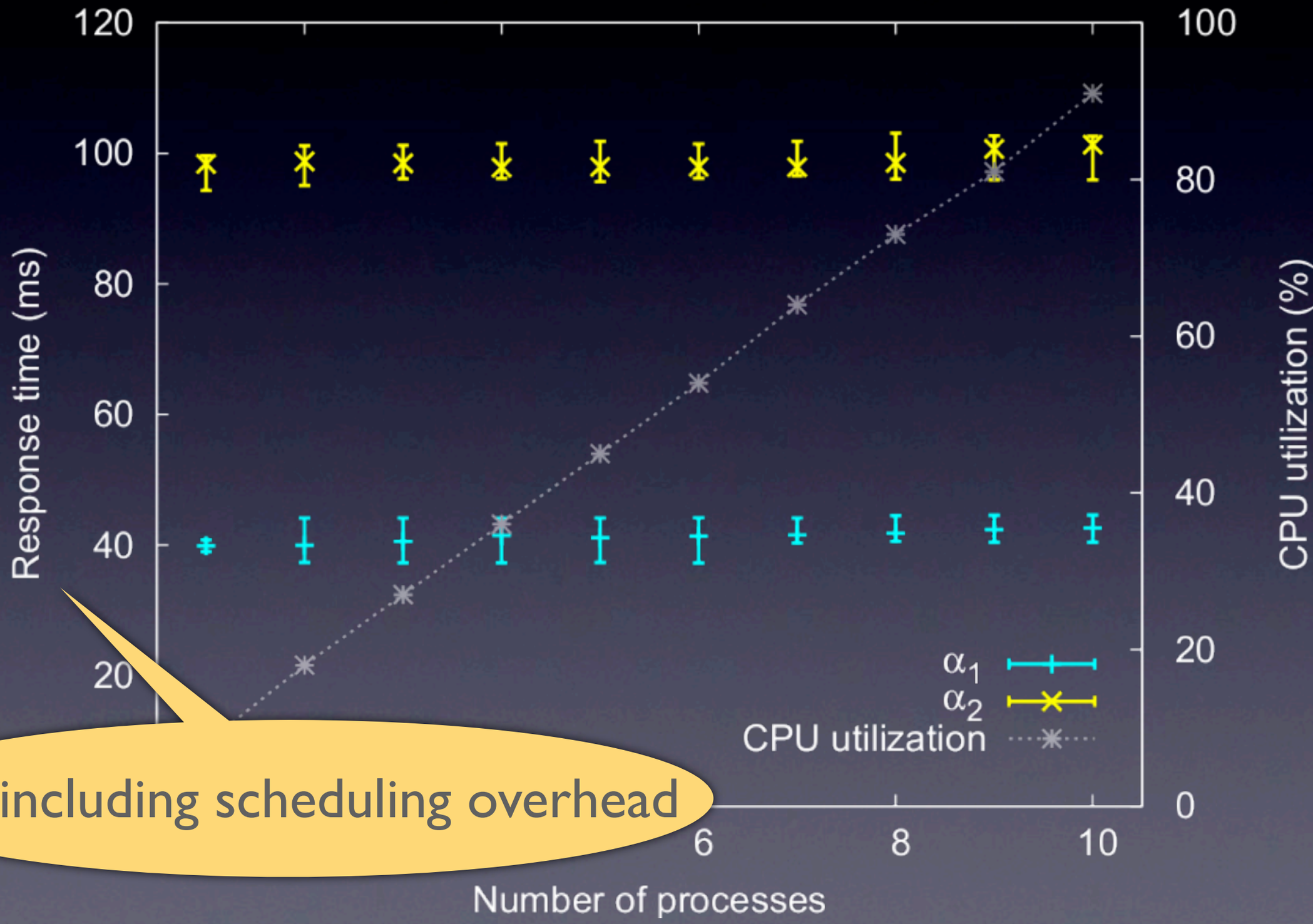
For any action α on a resource (λ, π) we have:

- upper response-time bound $\left\lceil \frac{load}{\lambda} \right\rceil \pi + \pi - 1$
- lower response-time bound $\left\lceil \frac{load}{\lambda} \right\rceil \pi$
- jitter $\pi - 1$

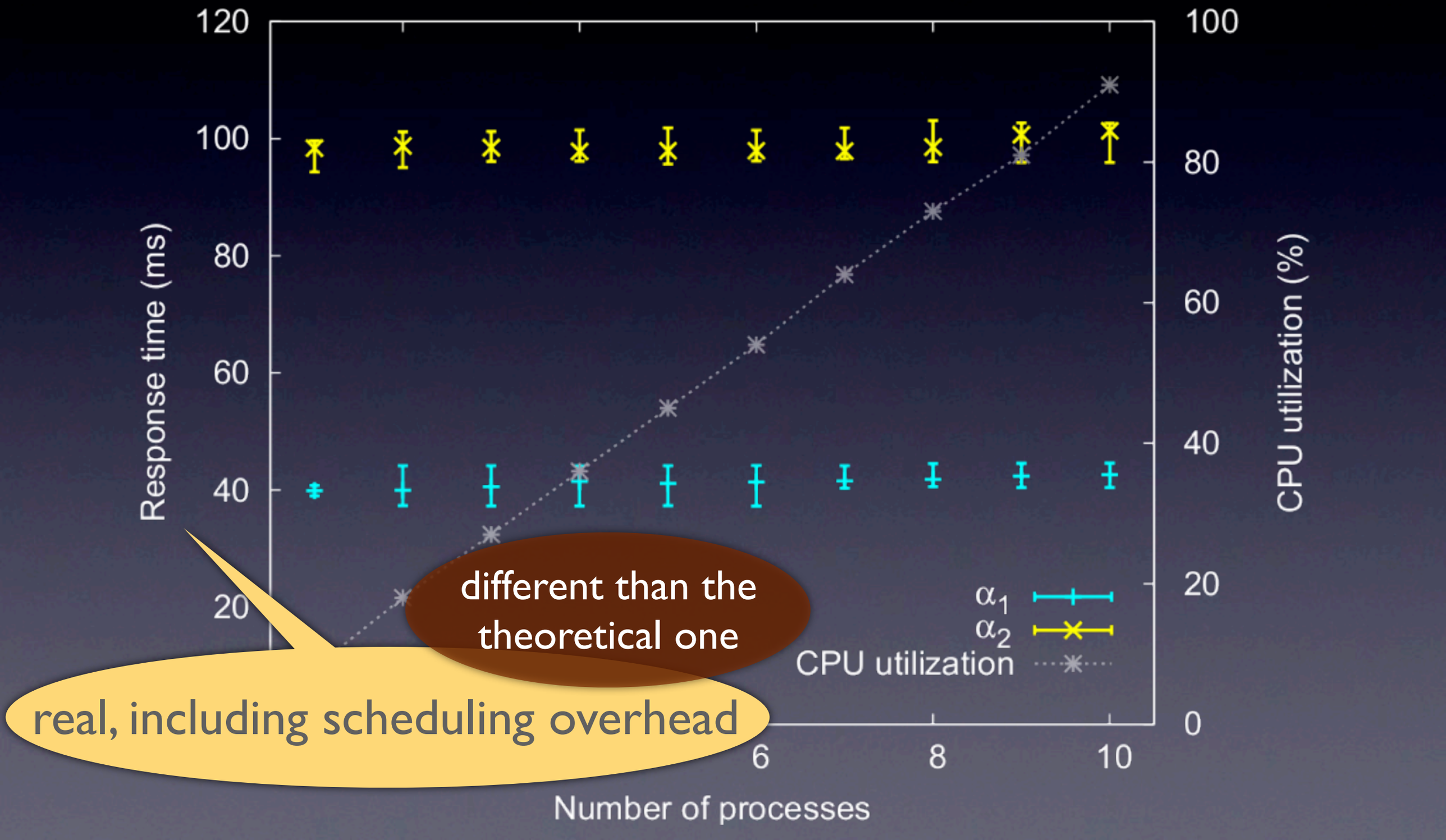
Motivating bare-metal experiment



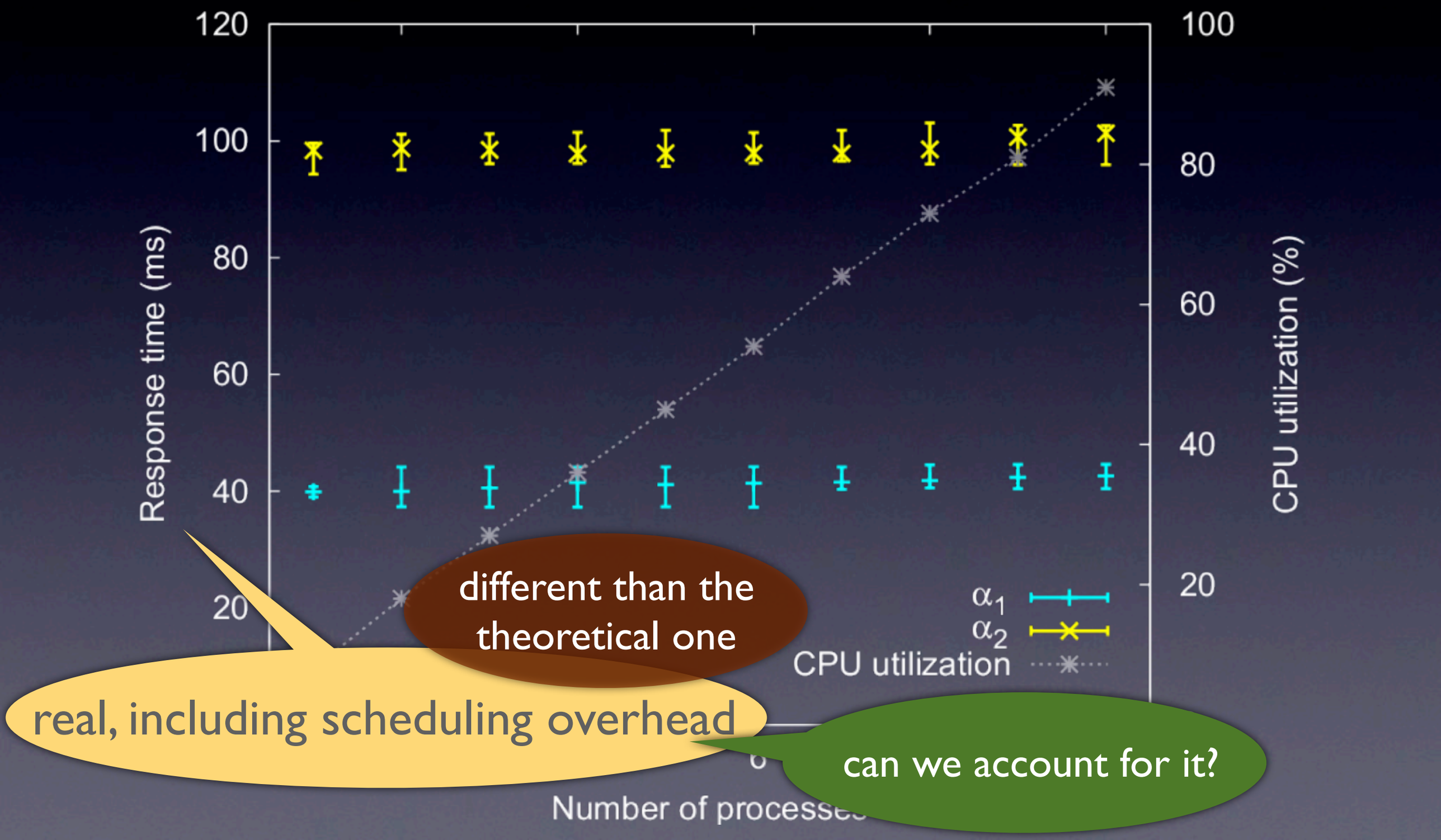
Motivating bare-metal experiment



Motivating bare-metal experiment



Motivating bare-metal experiment



Yes, we can

Provided a bound on the number of scheduler invocations is known

VBS, CBS,...

Yes, we can

Provided a bound on the number of scheduler invocations is known

VBS, CBS,...

Yes, we can

Provided a bound on the number of scheduler invocations is known

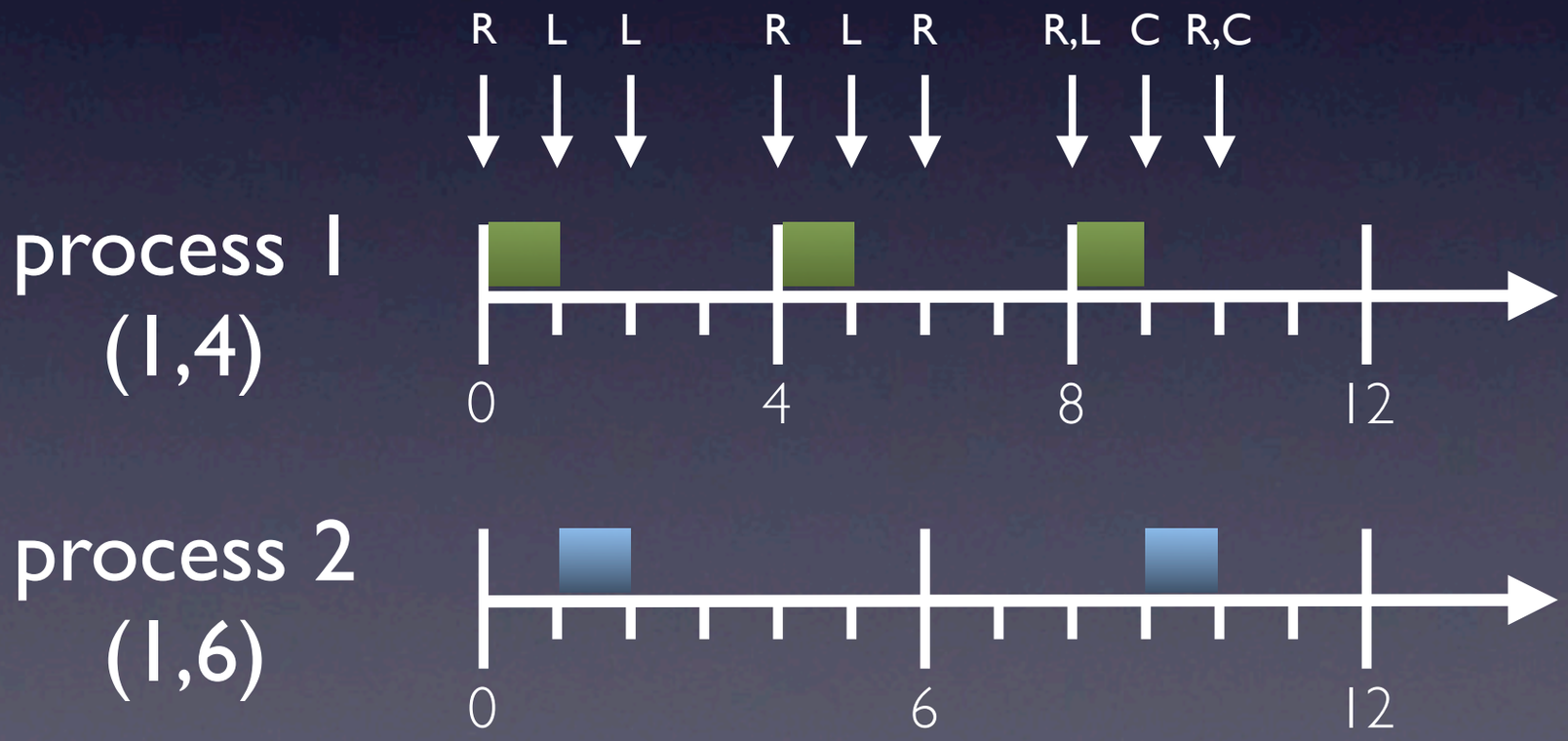
Reasons for scheduler invocation with VBS: release, limit/completion

VBS, CBS,...

Yes, we can

Provided a bound on the number of scheduler invocations is known

Reasons for scheduler invocation with VBS: release, limit/completion



Yes, we can

VBS, CBS,...

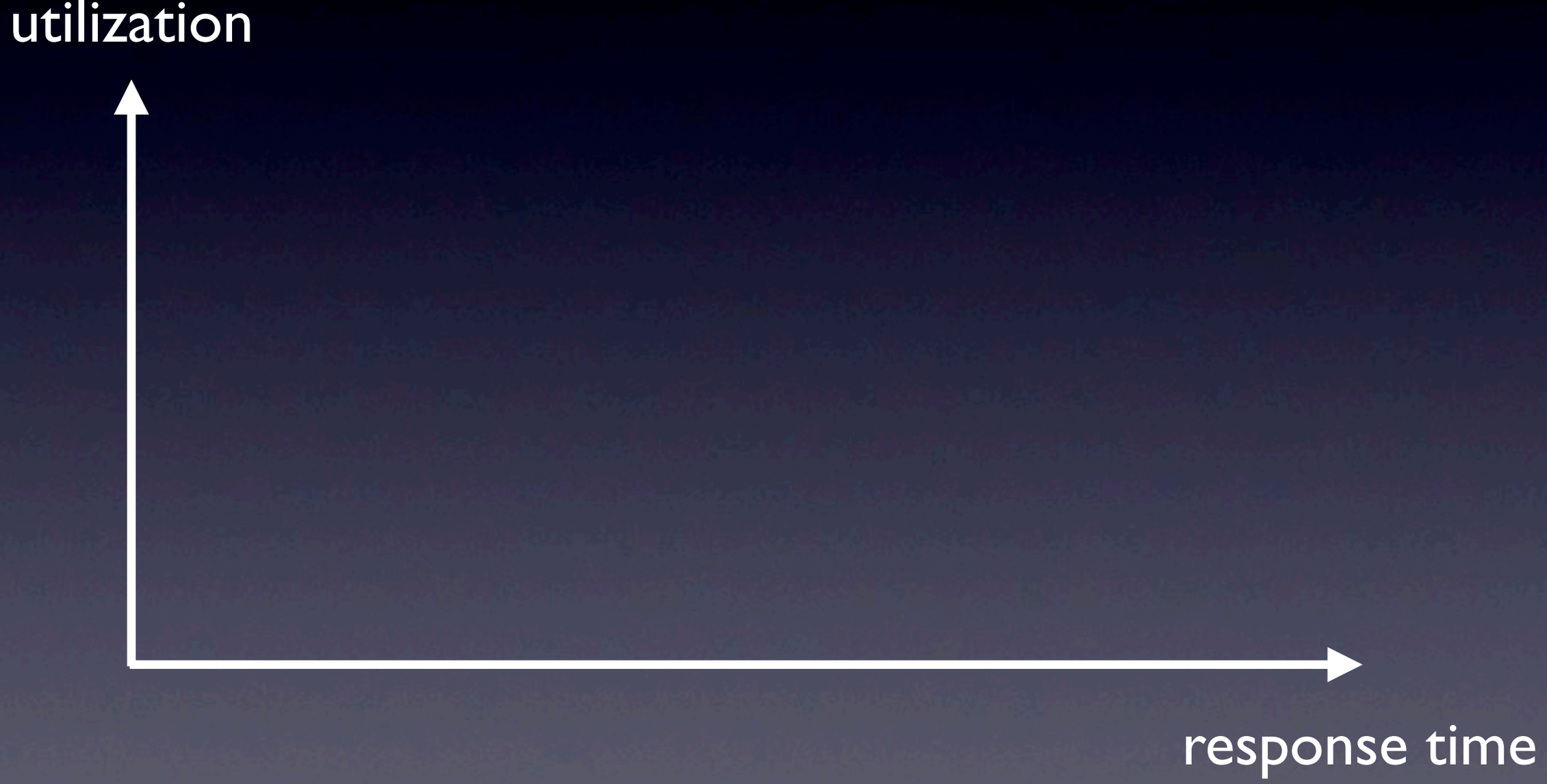
Provided a bound on the number of scheduler invocations is known

Reasons for scheduler invocation with VBS: release, limit/completion

for VBS

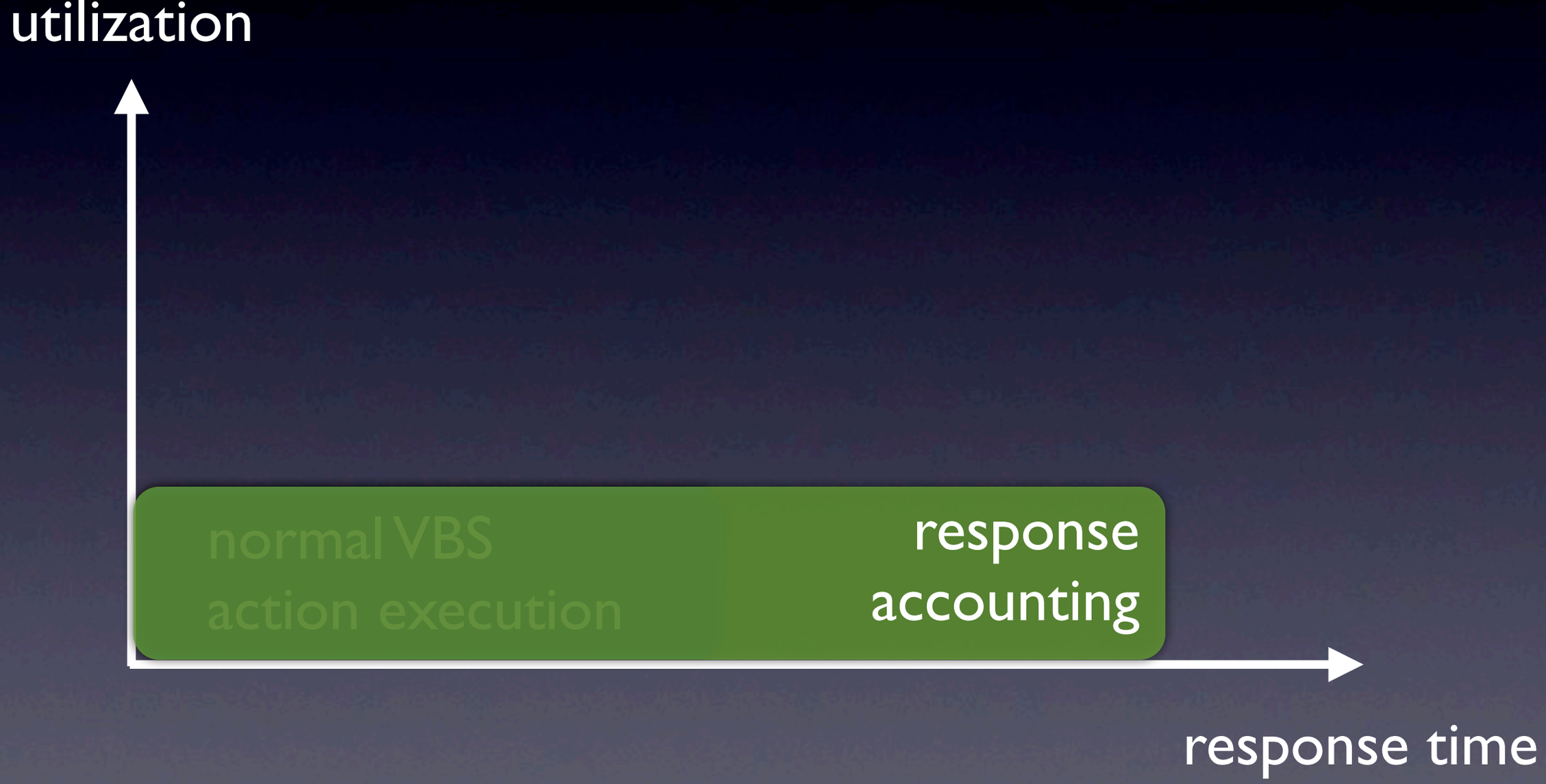
$$N = \left\lceil \frac{\pi}{\gcd(\text{all periods})} \right\rceil + 1$$

Overhead accounting

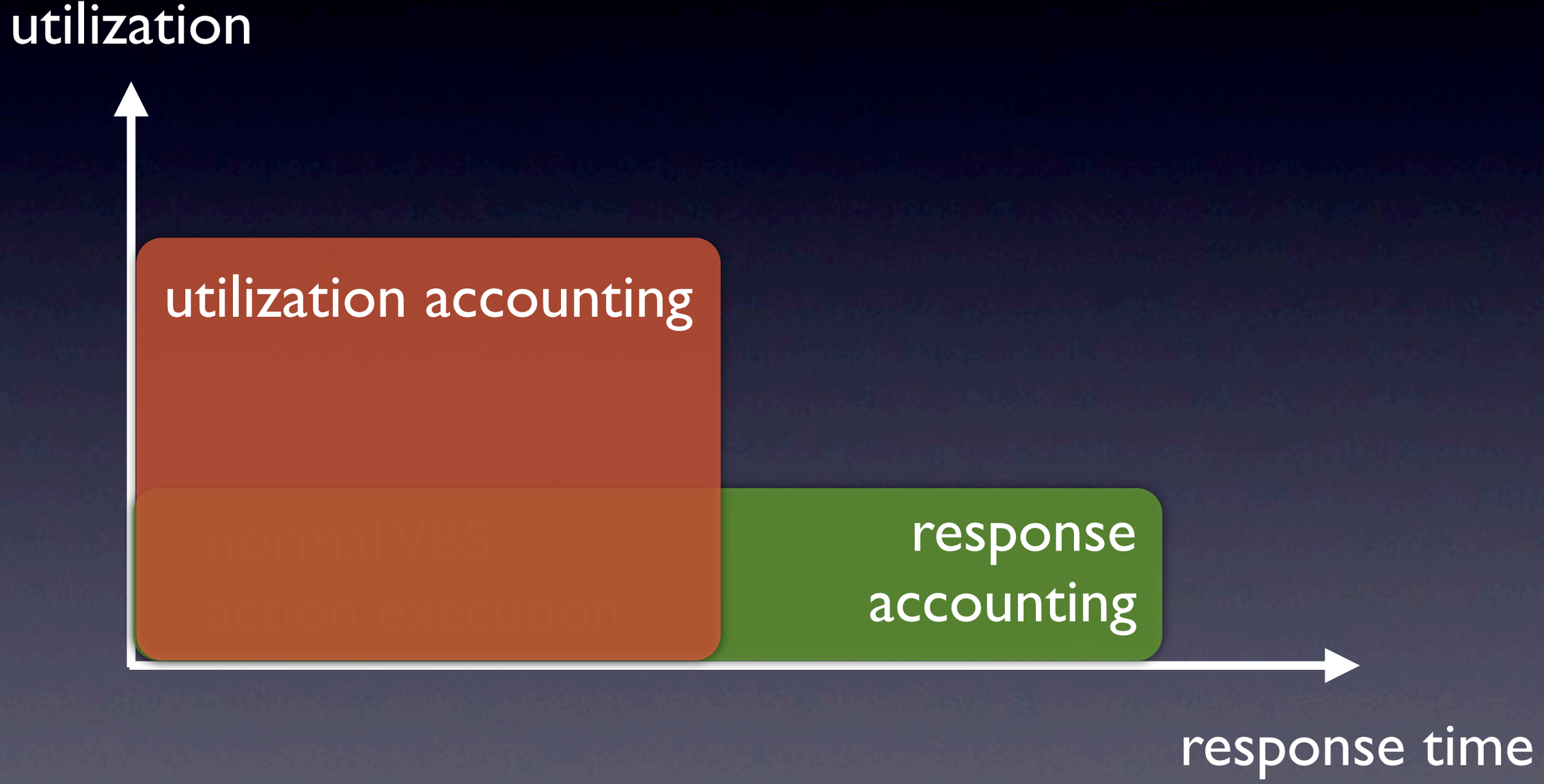


Overhead accounting

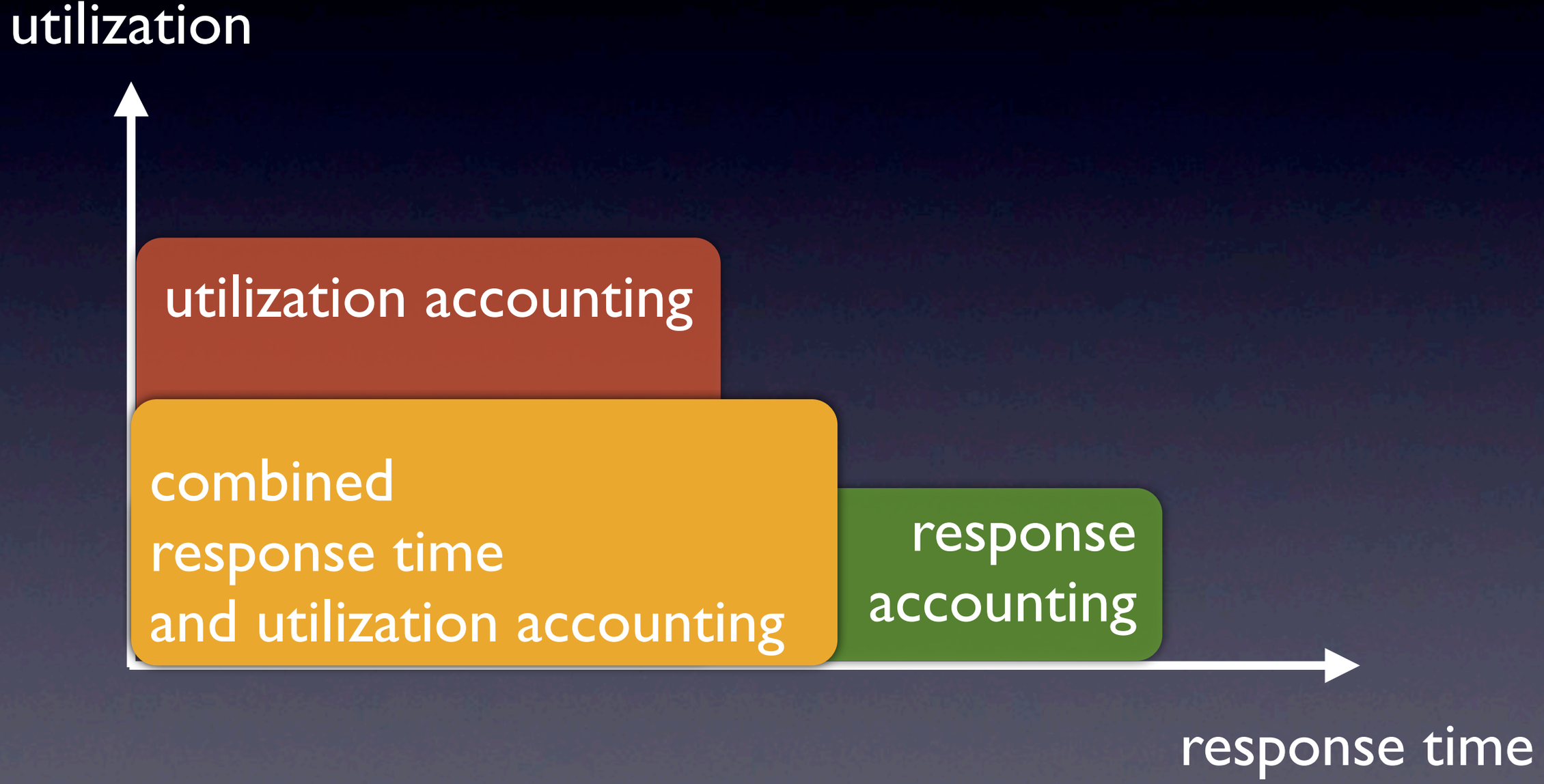
Overhead accounting



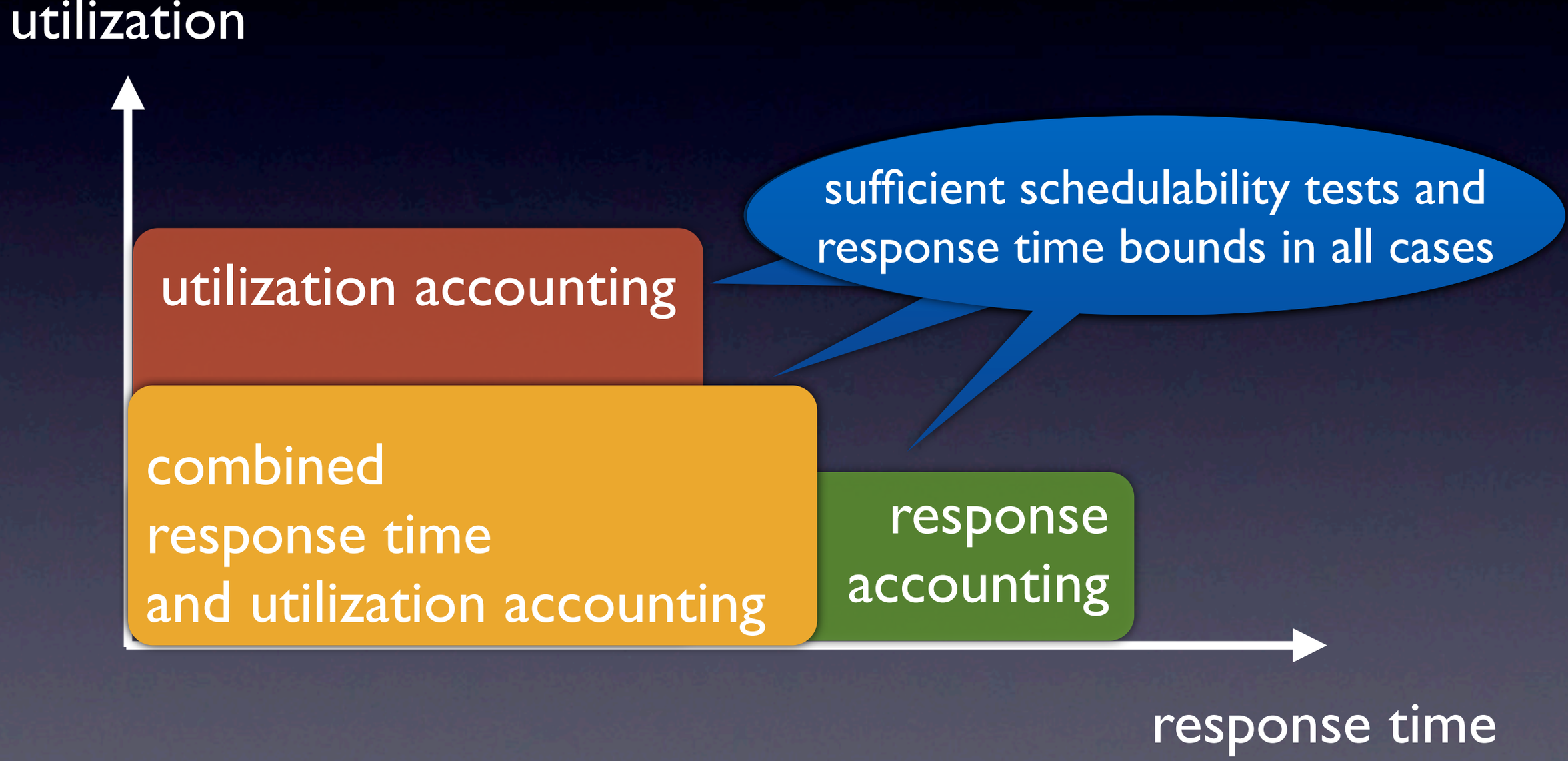
Overhead accounting



Overhead accounting



Overhead accounting



Without overhead

Without overhead

utilization

test: $\sum_i u_i \leq 1$

bounds:

$$\left\lceil \frac{\text{load}}{\lambda} \right\rceil \pi \leq \text{RT} \leq \left\lceil \frac{\text{load}}{\lambda} \right\rceil \pi + \pi - 1$$

normal VBS
action execution

response time

Response accounting

Response accounting

utilization

test: $\sum_i u_i \leq 1$

bounds:

$$\left\lceil \frac{\text{load}^*}{\lambda} \right\rceil \pi \leq \text{RT} \leq \left\lceil \frac{\text{load}^*}{\lambda} \right\rceil \pi + \pi - 1$$

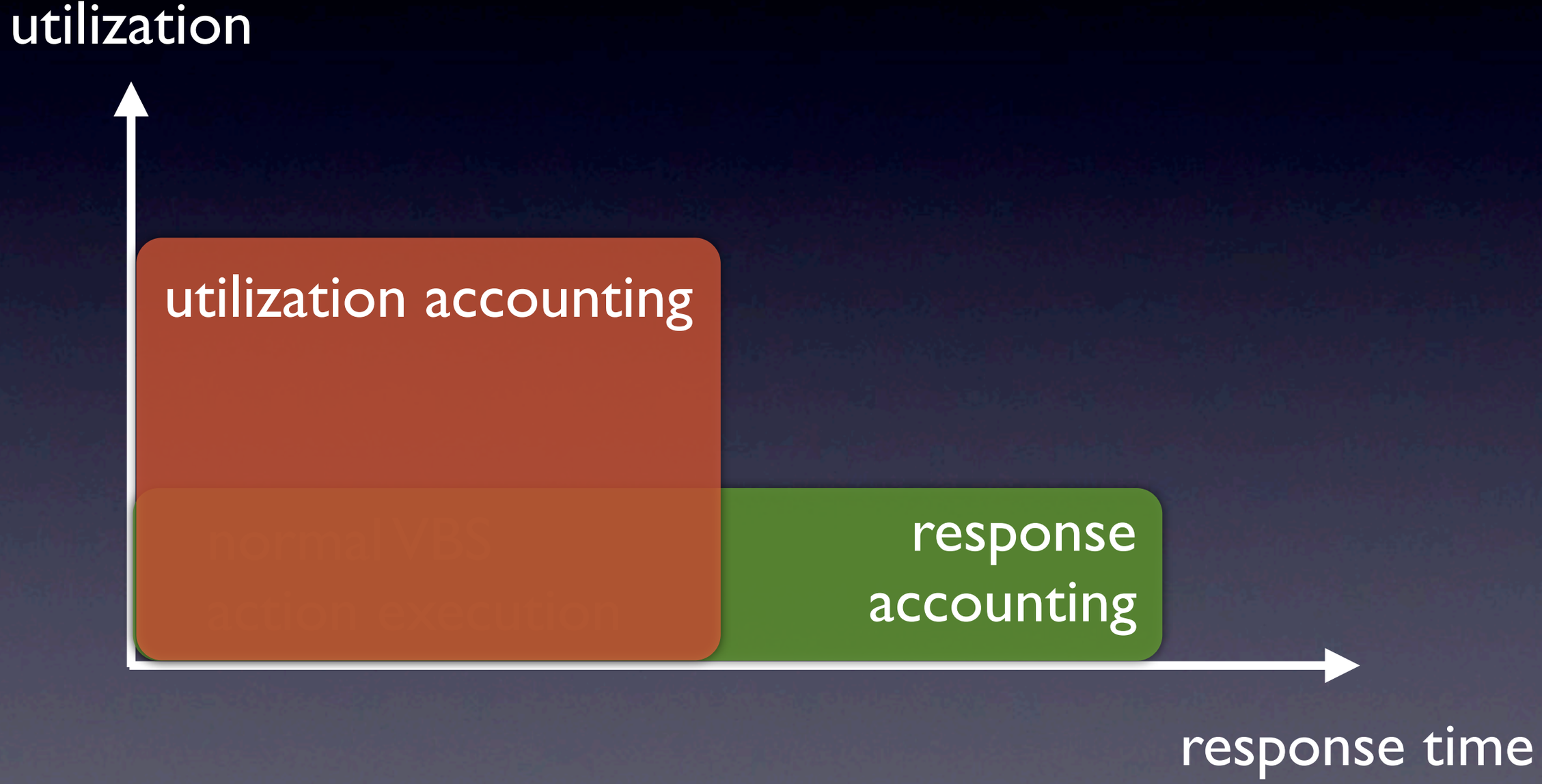
$$\text{load} + \left\lceil \frac{\text{load}}{\lambda - \delta} \right\rceil \delta$$

normal VBS action execution

response accounting

response time

Utilization accounting

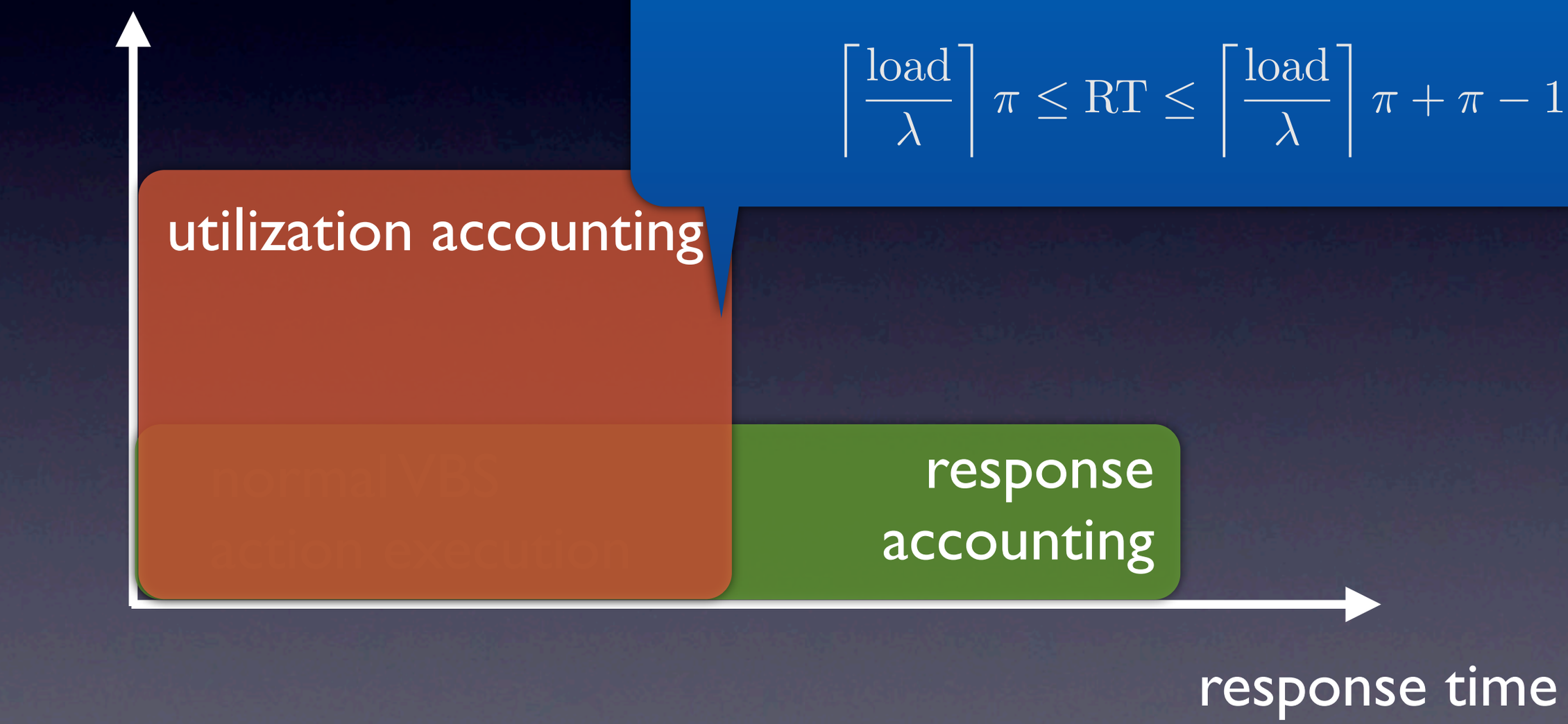


Utilization accounting

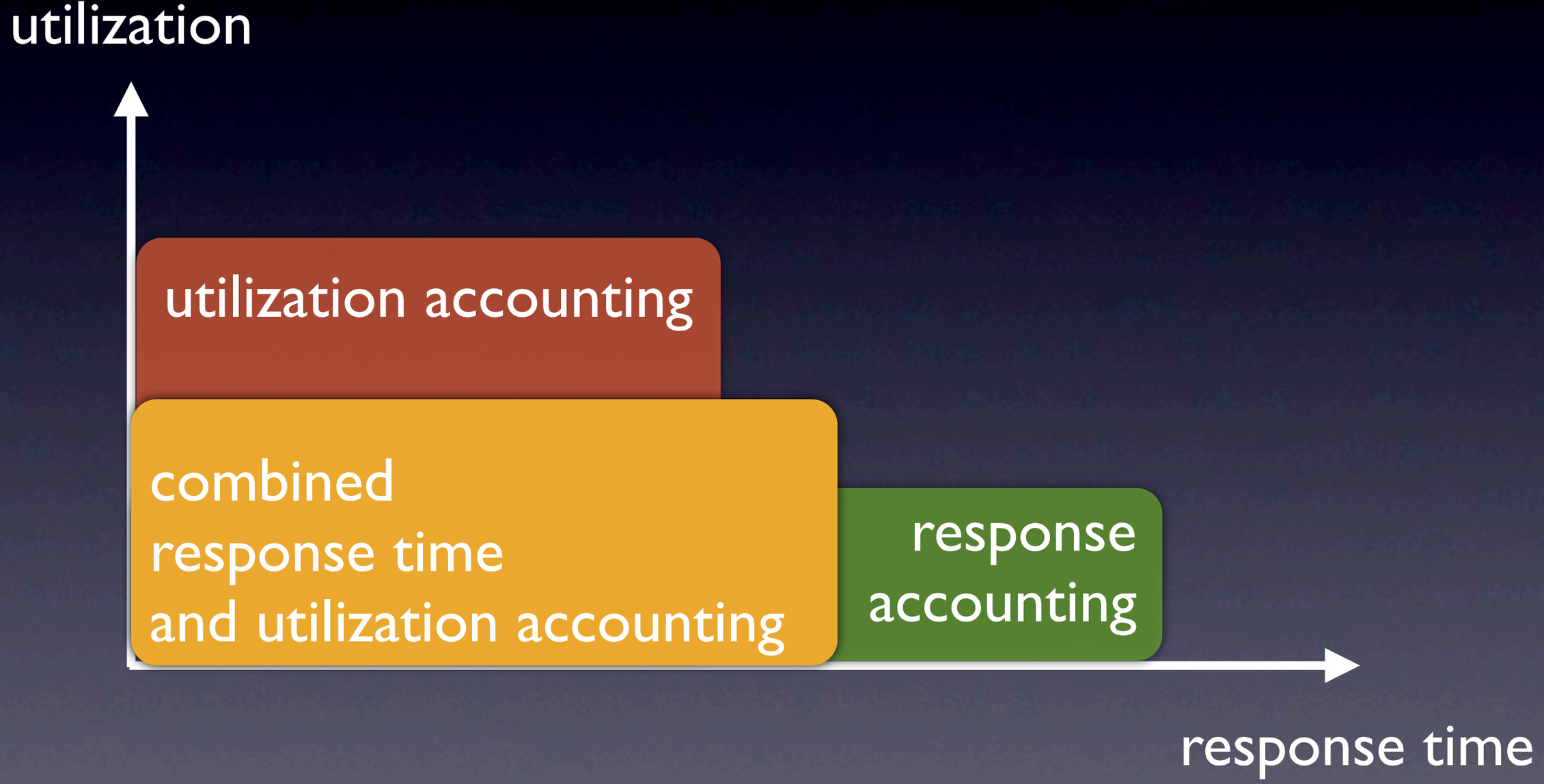
test: $\sum_i \max_j \frac{\lambda_{i,j} + \delta_{i,j}}{\pi_{i,j}} \leq 1$

bounds:

$$\left\lceil \frac{\text{load}}{\lambda} \right\rceil \pi \leq \text{RT} \leq \left\lceil \frac{\text{load}}{\lambda} \right\rceil \pi + \pi - 1$$



Utilization accounting



Utilization accounting

test: $\sum_i \max_j \frac{\lambda_{i,j} + \delta^u_{i,j}}{\pi_{i,j}} \leq 1$

bounds:

$$\left\lceil \frac{\text{load}^*}{\lambda} \right\rceil \pi \leq \text{RT} \leq \left\lceil \frac{\text{load}^*}{\lambda} \right\rceil \pi + \pi - 1$$

$\text{load}' + \left\lceil \frac{\text{load}'}{\lambda} \right\rceil \delta^u$

$\text{load} + \left\lceil \frac{\text{load}}{\lambda - \delta^b} \right\rceil \delta^b$

$\delta = \delta^u + \delta^b$

utilization

utilization accounting

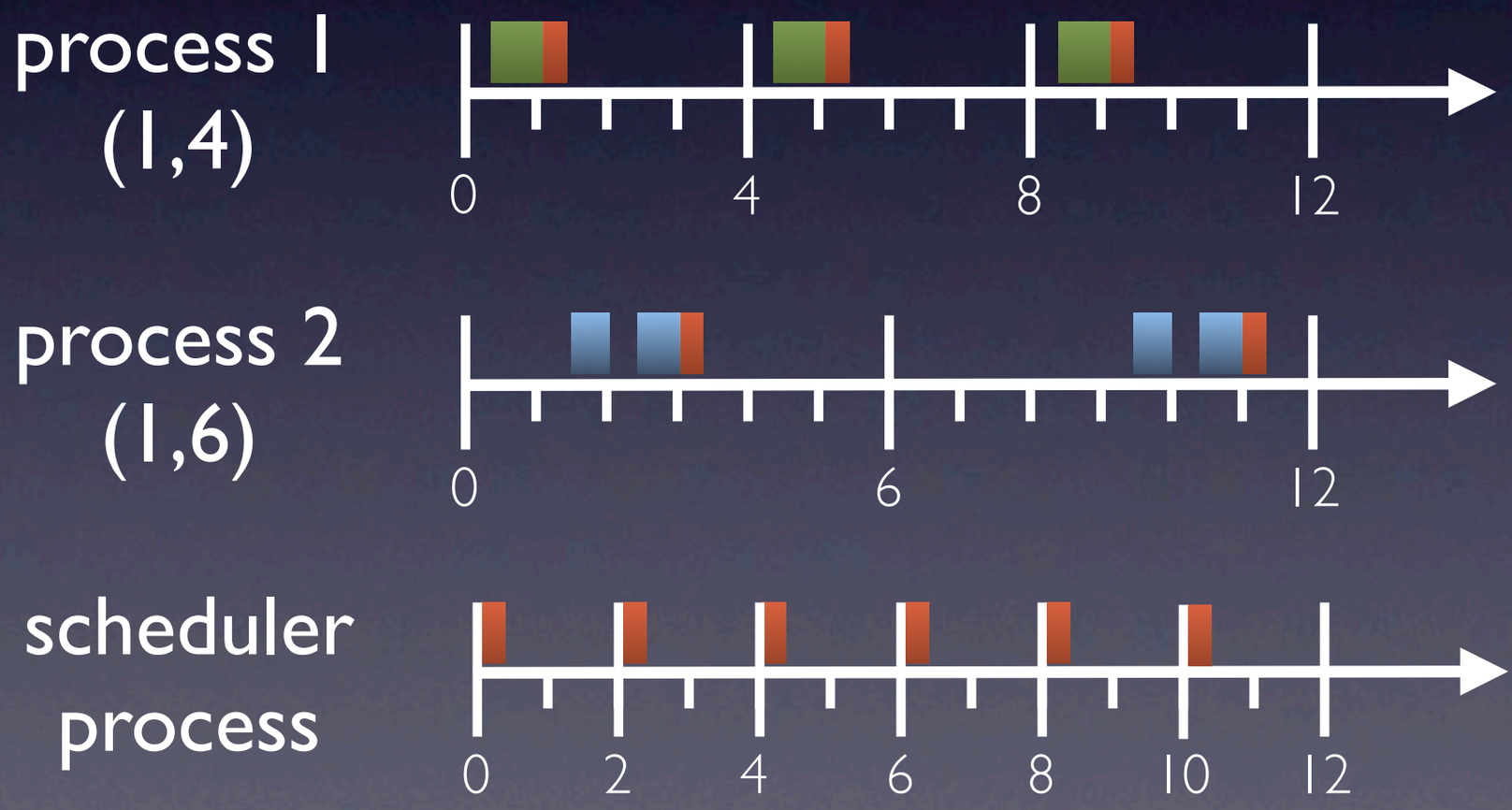
combined
response time
and utilization accounting

response
accounting

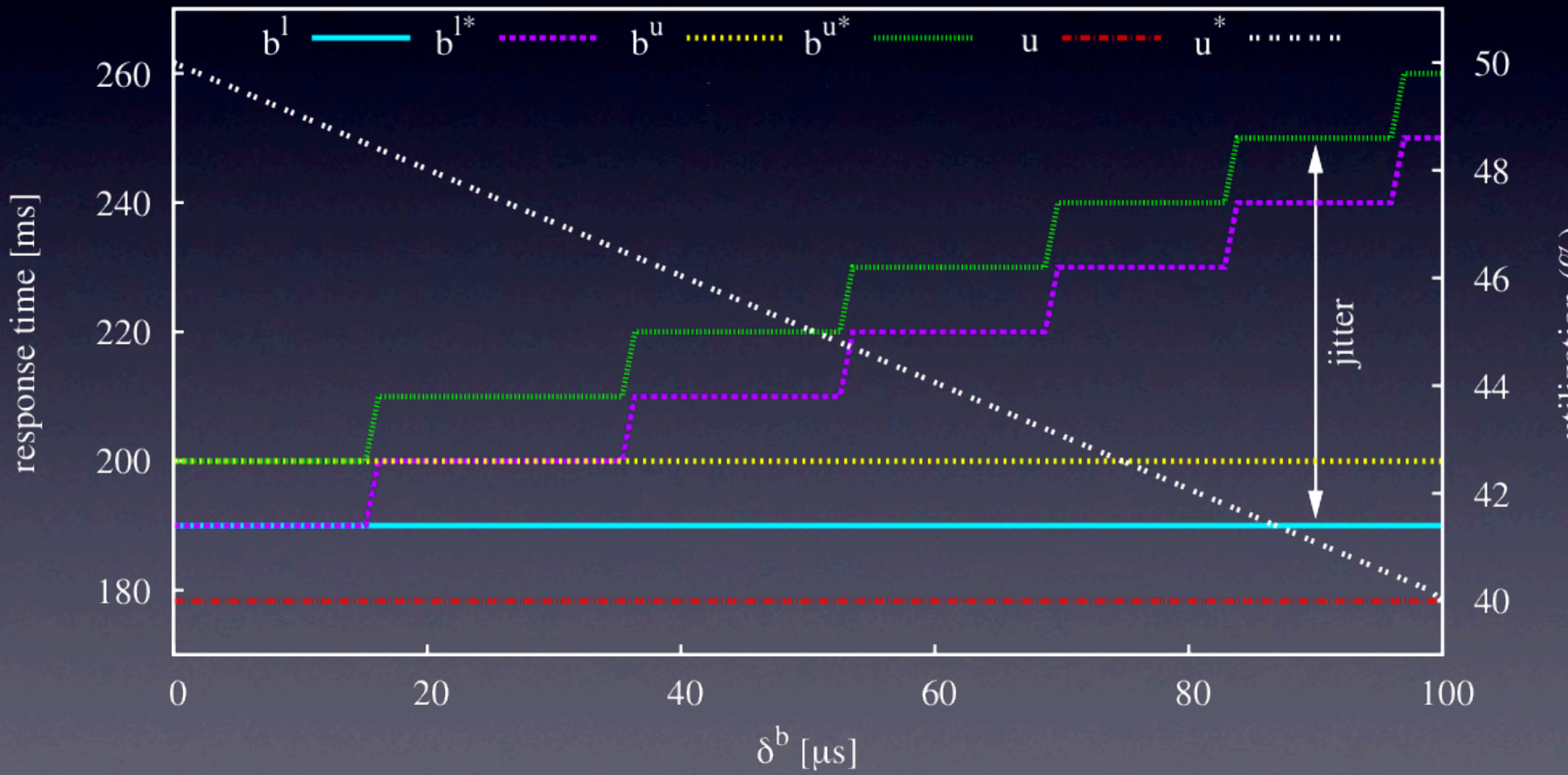
response time

Possible optimization

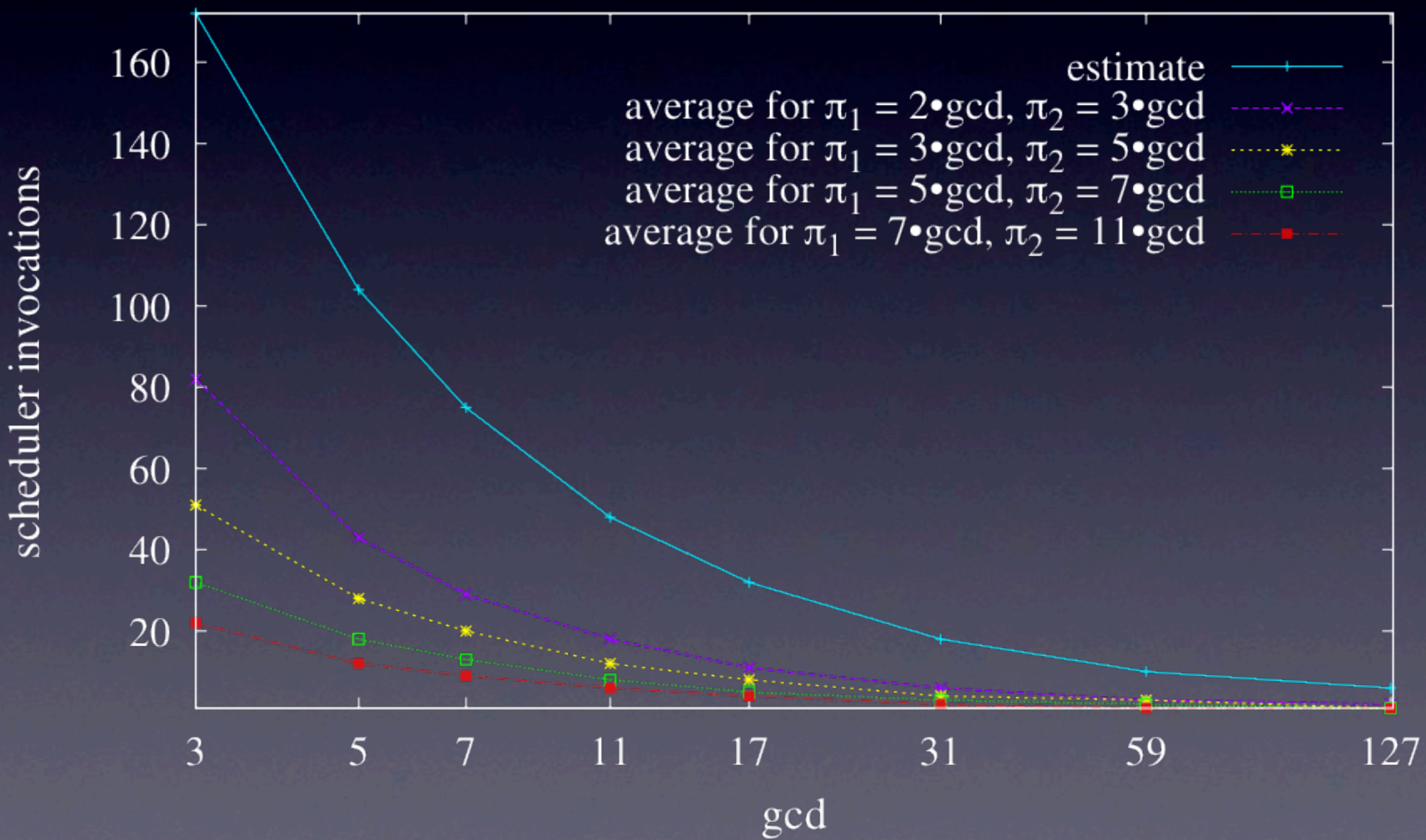
Scheduling invocations due to release can be considered as a separate process



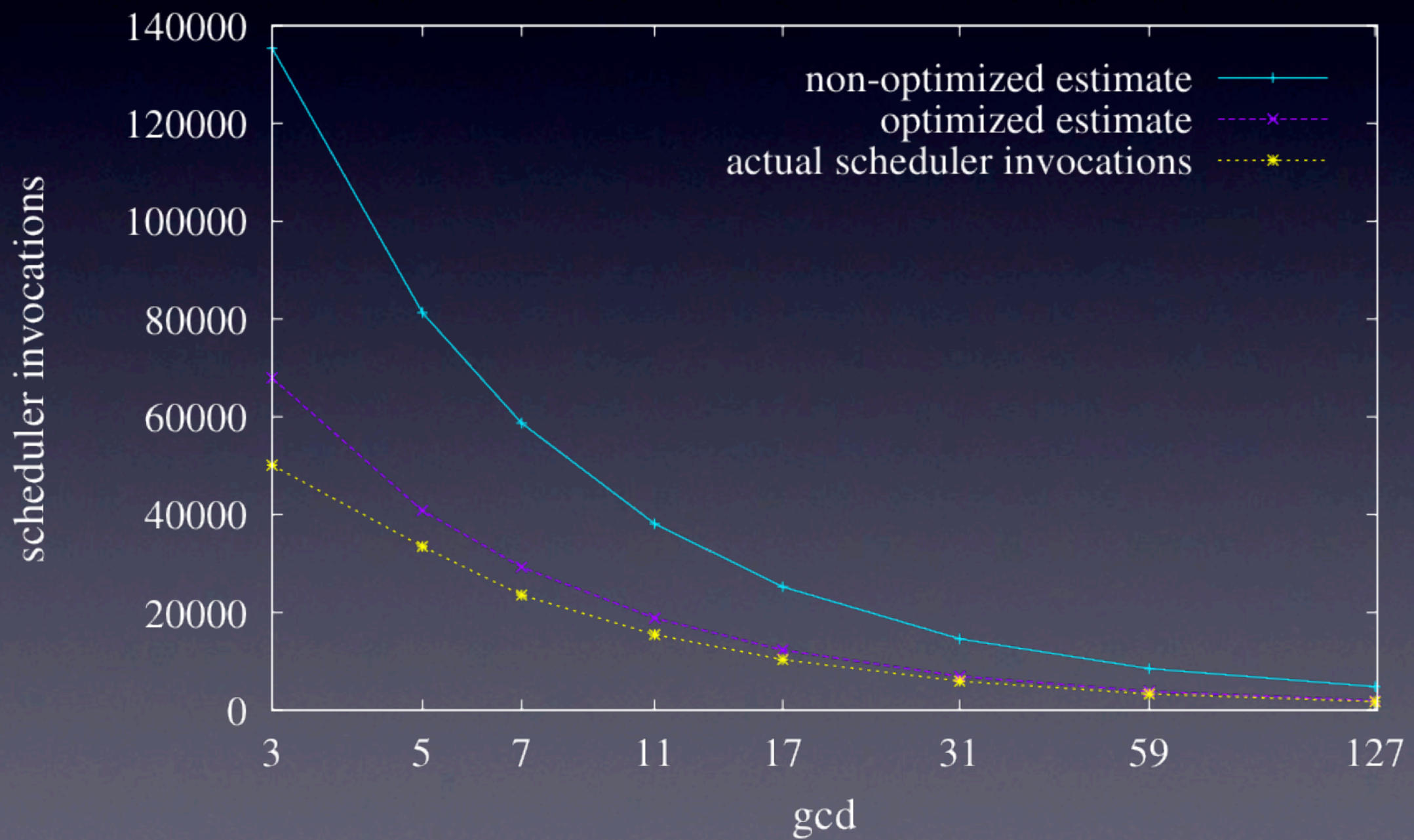
Overhead accounting



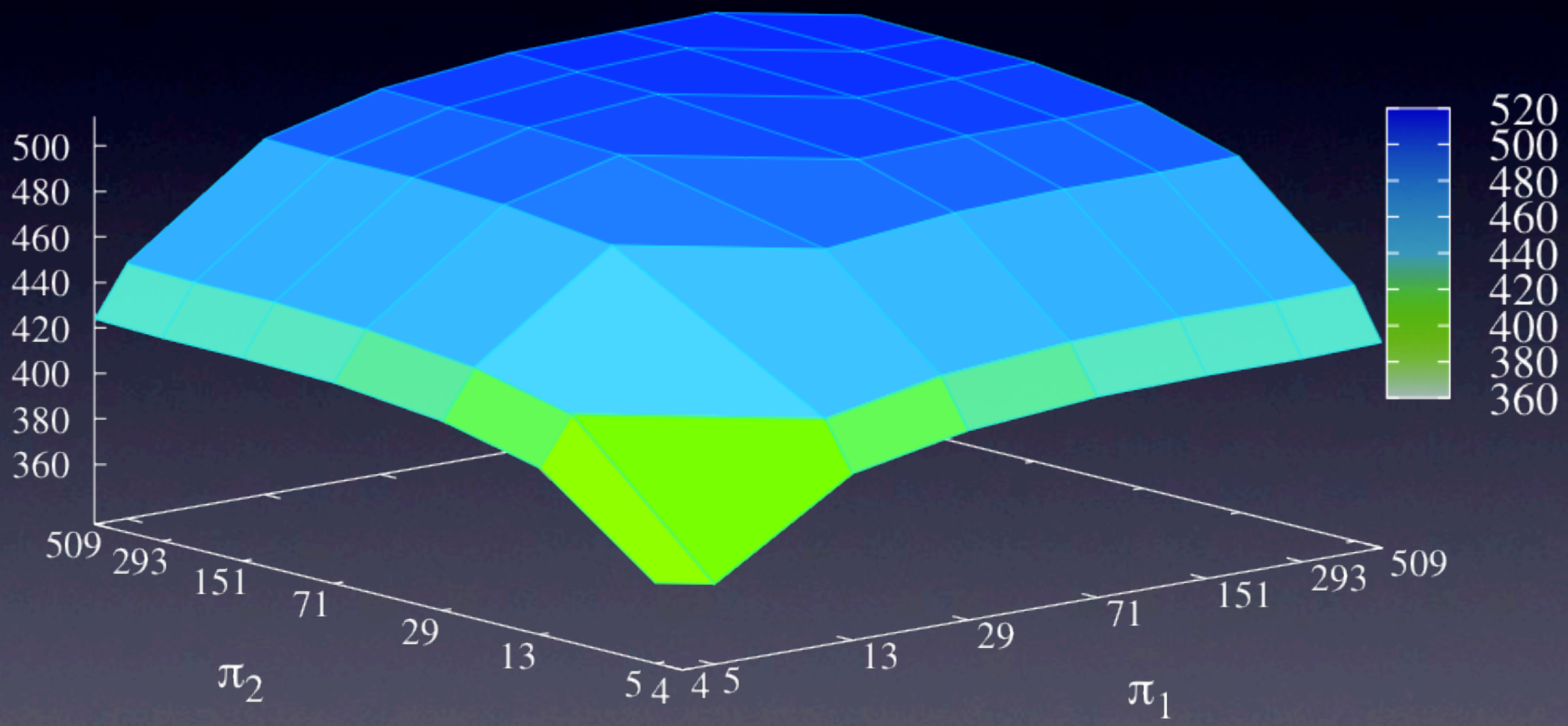
Experiments



Experiments



Experiments

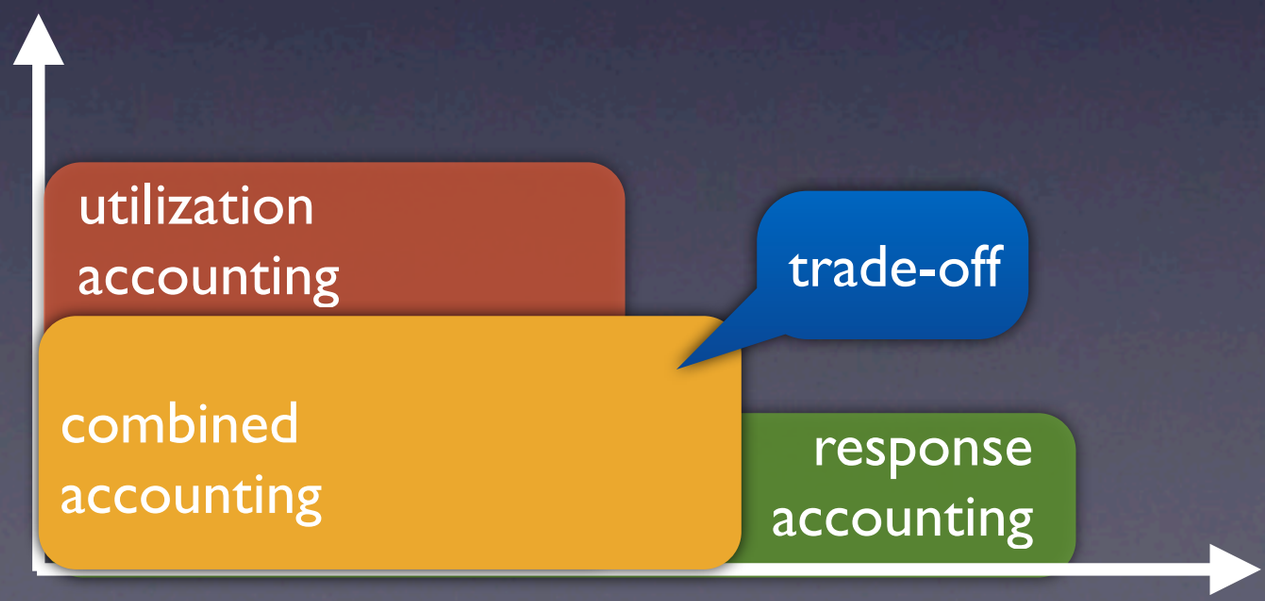


Conclusions

- Reservation-based scheduling (CBS, VBS, ..) allows for temporal isolation and scheduler overhead accounting

- Bound on scheduler invocations

- As a result:

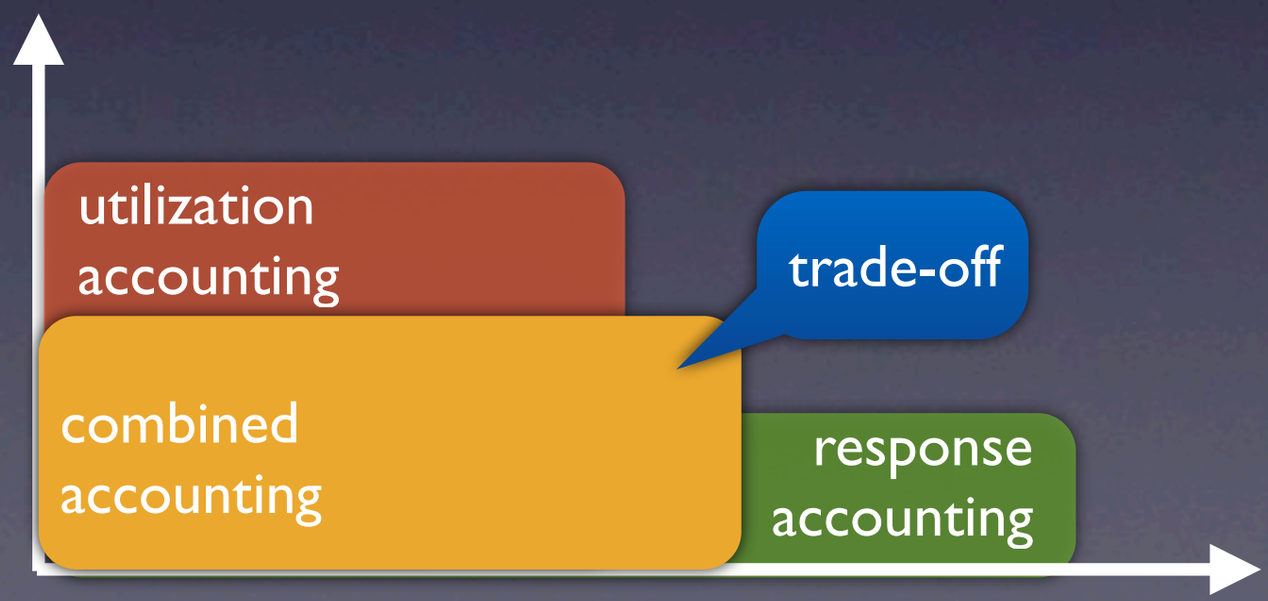


Conclusions

- Reservation-based scheduling (CBS, VBS, ..) allows for temporal isolation and scheduler overhead accounting

- Bound on scheduler invocations

- As a result:



Thank you