Universität Bremen

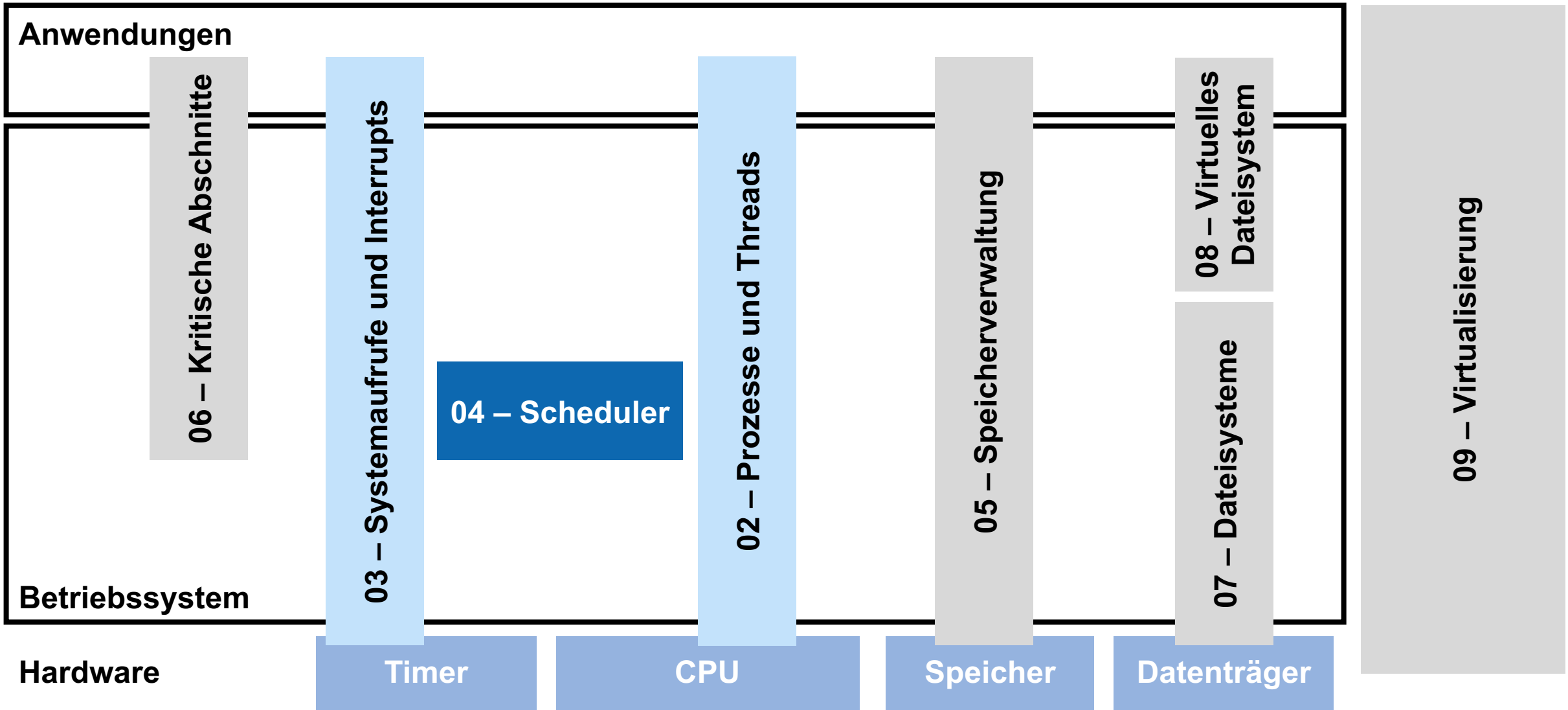
Betriebssysteme

Scheduler

Dr. Bernhard J. Berger
AG Sichere Systeme

Universität Bremen

Themenübersicht



Universität Bremen

Themenübersicht

- Rückblick
- Anforderungen
- Zeitpunkt
- Beispiel: Offline
- Dynamische Verfahren
- Auswirkung auf Durchsatz

Scheduling

- Metriken
  - Deskriptiv
  - Wertend
- Prinzipien
  - First Come, First Serve
  - Round Robin
  - Earliest Deadline First
  - Fair Scheduling

Algorithmen

- Priority Scheduler
- Datenstrukturen
- Prioritäten
- Algorithmen

O(1) Priority-Scheduler

Erinnerung: Prozesswechsel

- Aktiver Prozess: Kontext im Prozesskontrollblock speichern
  - Registerwerte
  - Aktualisierung des Zustands (Ready vs. Blocked)
- Nächster Prozess wird bestimmt (u.U. Idle-Prozess)
- Neuer Prozess: Wiederherstellen des Kontextes
  - Registerwerte wiederherstellen
  - Aktualisierung des Zustands (Running)

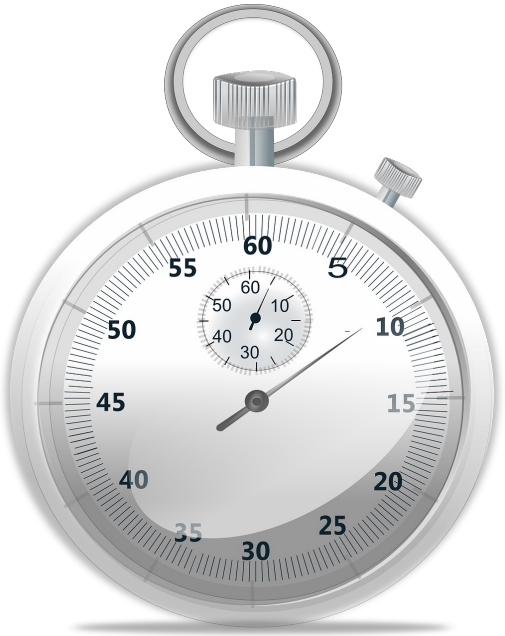
Definition

Der Dispatcher verwaltet die CPUs eines Systems und aktiviert und deaktiviert Prozesse. Die Auswahl des aktiven Prozesses und des Prozesswechsels wird als Scheduling bezeichnet.

Anforderungen

- **Scheduling ist ein Optimierungsproblem:**
- Usability
  - Reaktionszeit der Prozesse
  - QoS: Zuordnung nach *Bezahlmodell*
  - Fairness (Gleichverteilung der CPU)
- Durchsatz
  - Dauer bis zum Abschluss
  - CPU-Auslastung
- Safety
  - Garantierte maximale Reaktionszeit
  - Ausschluss des *Verhungerns*

- **Neben CPU-Verfügbarkeit**
  - Verfügbarkeit von IO-Zugriff
  - Verfügbarkeit von Arbeitsspeicher



Hinweis

Es gibt keinen idealen Scheduler für alle Situationen. Daher bieten Betriebssysteme häufig mehrere Implementierung an.

Zeitpunkt

Offline-Scheduling

- Planung vor Ausführung
- Statische Ressourcenverteilung
- Scheduling-Eigenschaften nachweisbar
  - Vorteil für Echtzeitsysteme
  - Erfordert Kenntnis über Prozesse/eigenschaften
- Erlaubt sehr gute Auslastung
- Unterschiedliche Scheduling-Pläne möglich
  - Plan abhängig von Zustand
  - Vorsicht bei Zustandswechsel, ob Anforderungen eingehalten werden können
- Häufig eingesetzt bei spezieller Hardware
  - Motorsteuerungen
  - Echtzeitsysteme

Online-Scheduling

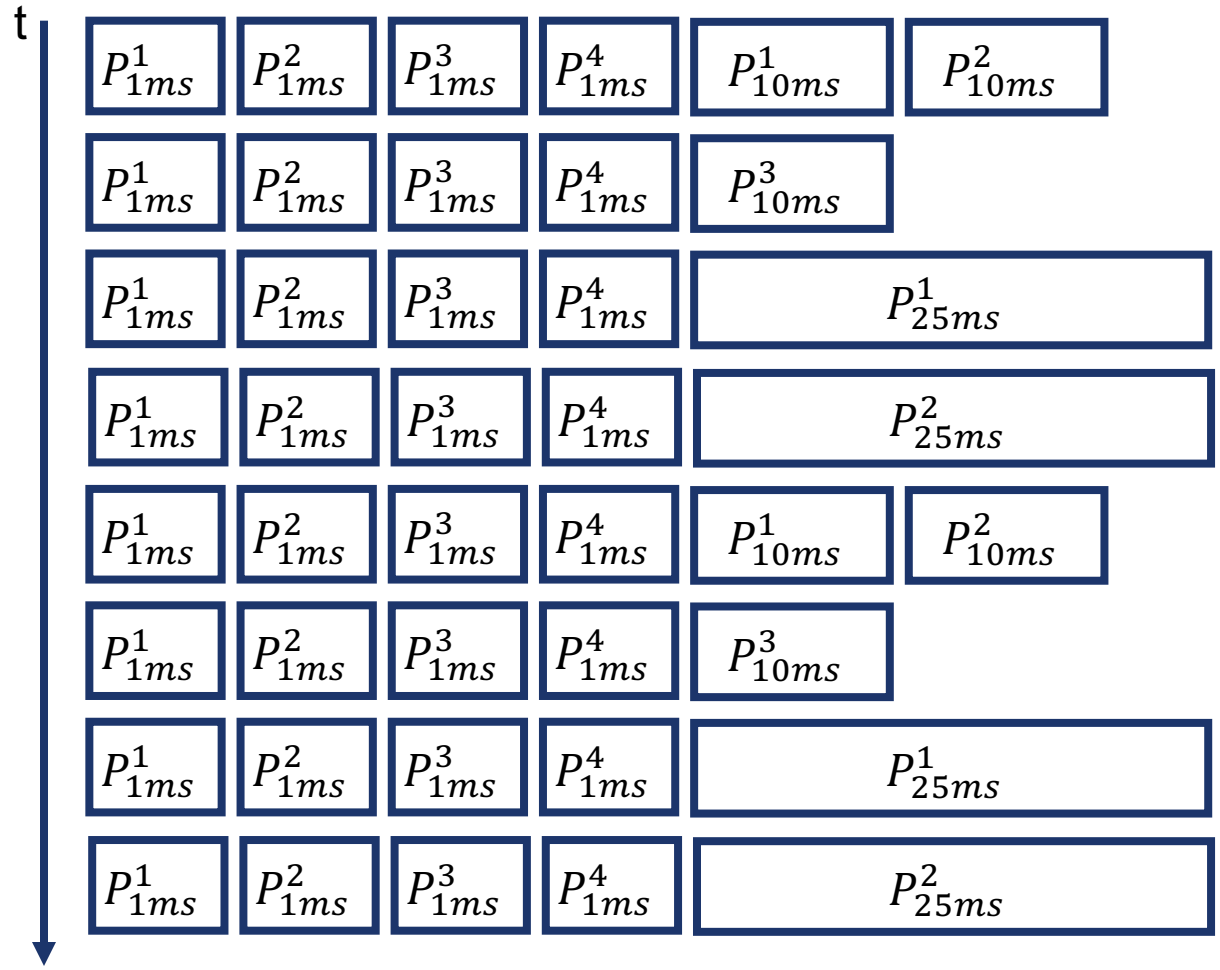
- Planung während Ausführung
- Dynamische Ressourcenverteilung
- Verursacht Planungs-Overhead
  - Verringert Durchsatz
  - Reduziert Optimalität
  - Balance wiederum ein Optimierungsproblem
- Prozesse können Prioritäten zugewiesen werden
- Erlaubt Reaktion auf dynamische Events
  - Umplanung möglich
  - Verbessert Auslastung
- Abwägung führt zu unterschiedlichen Schedulern

Beispiel: Offline-Scheduling

- Tasks werden nach Laufzeit klassifiziert

  Beispiel
  - P_{1ms}: Prozesse mit weniger als $1ms$ Laufzeit
  - P_{10ms}: Prozesse mit weniger als $10ms$ Laufzeit
  - P_{25ms}: Prozesse mit weniger als $25ms$ Laufzeit
- Anforderung:
  - Prozesse der Gruppe P_{1ms} müssen $\leq 30ms$ reagieren können (z.B Airbag)
  - $|P_{1ms}| = 4$
- Scheduler-Umsetzung
  - Virtuelle Zeitslots von $30ms$
  - Führe alle Prozesse aus P_{1ms} aus
  - Fülle Zeitslots auf aus restlichen Slots auf

  Wichtig: Gesamtlaufzeit $\leq 30ms$



Dynamische Verfahren

Nicht-Präemptives Scheduling

- Prozess wird von OS aktiviert (erhält CPU)
- Aktiver Prozess bestimmt Prozesswechsel
  - Rechnet bis zum Ende
  - Oder gibt CPU frei

Präemptives Scheduling

- Betriebssystem unterbricht aktiven Prozess
- Betriebssystem aktiviert Prozess später wieder
- Scheduler entscheidet über Zuteilung
- Ermöglicht Prozesspriorisierung

Hinweis

Präemtives Scheduling wird mit Hilfe des Timers (siehe letzte Vorlesung) umgesetzt. Dieser erlaubt es den aktuellen Prozess zu verdrängen und das Scheduling durchzuführen

Universität Bremen

Themenübersicht

- Rückblick
- Anforderungen
- Zeitpunkt
- Beispiel: Offline
- Dynamische Verfahren
- Auswirkung auf Durchsatz

Scheduling

- Metriken
  - Deskriptiv
  - Wertend
- Prinzipien
  - First Come, First Serve
  - Round Robin
  - Earliest Deadline First
  - Fair Scheduling

Algorithmen

- Priority Scheduler
- Datenstrukturen
- Prioritäten
- Algorithmen

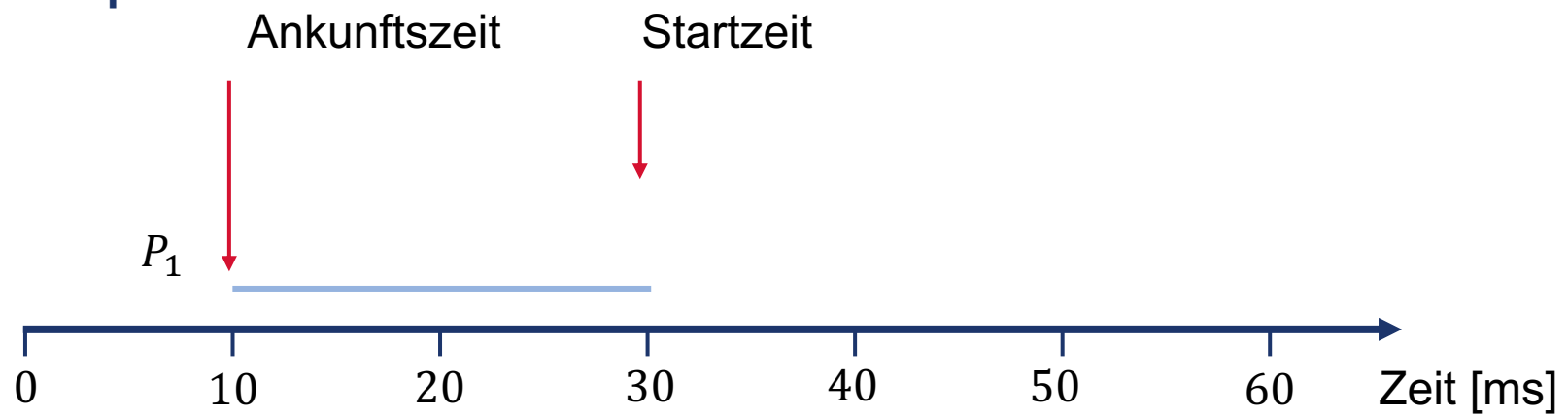
O(1) Priority-Scheduler

Metriken zur Prozessbeschreibung

- **Ankunftszeit** (arrival time): Zeitpunkt, an dem ein Prozess erzeugt wurde
- **CPU-Arbeitszeit** (burst time): Zeit, die der Prozess insgesamt rechnet

- **Startzeit** (starting time): Zeitpunkt, an dem ein Prozess zum ersten Mal die CPU nutzt
- **Wartezeit** (waiting time): Zeit, die ein Prozess in der Warteschlange verbringt
- **Endzeit** (finishing time): Zeitpunkt, an dem ein Prozess abgeschlossen ist

Beispiel



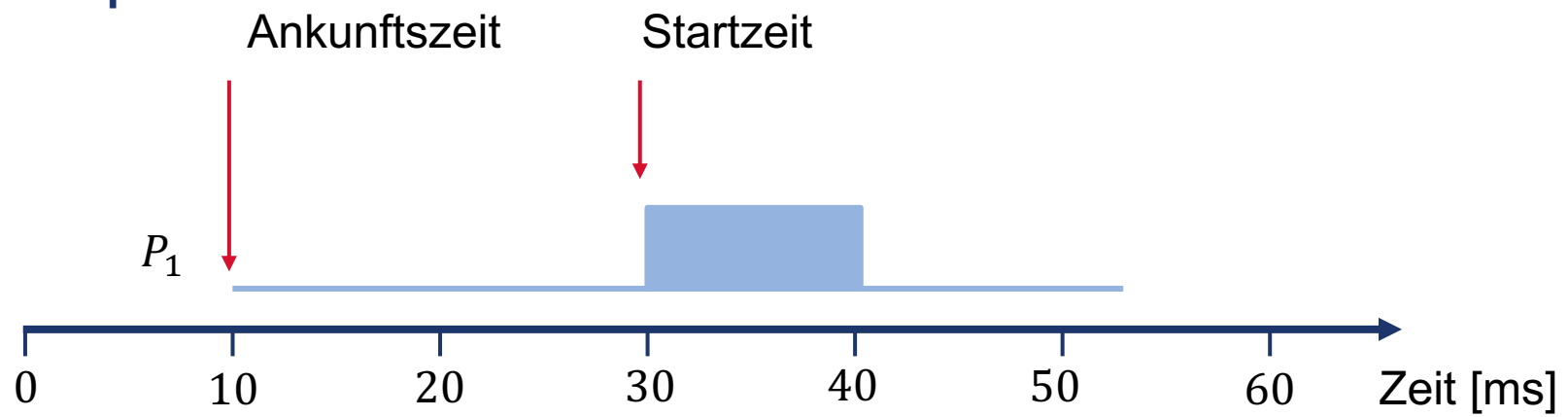
	P_1 [ms]
Ankunftszeit	10
Arbeitszeit	17
Startzeit	
Wartezeit	
Endzeit	

Metriken zur Prozessbeschreibung

- **Ankunftszeit** (arrival time): Zeitpunkt, an dem ein Prozess erzeugt wurde
- **CPU-Arbeitszeit** (burst time): Zeit, die der Prozess insgesamt rechnet

- **Startzeit** (starting time): Zeitpunkt, an dem ein Prozess zum ersten Mal die CPU nutzt
- **Wartezeit** (waiting time): Zeit, die ein Prozess in der Warteschlange verbringt
- **Endzeit** (finishing time): Zeitpunkt, an dem ein Prozess abgeschlossen ist

Beispiel



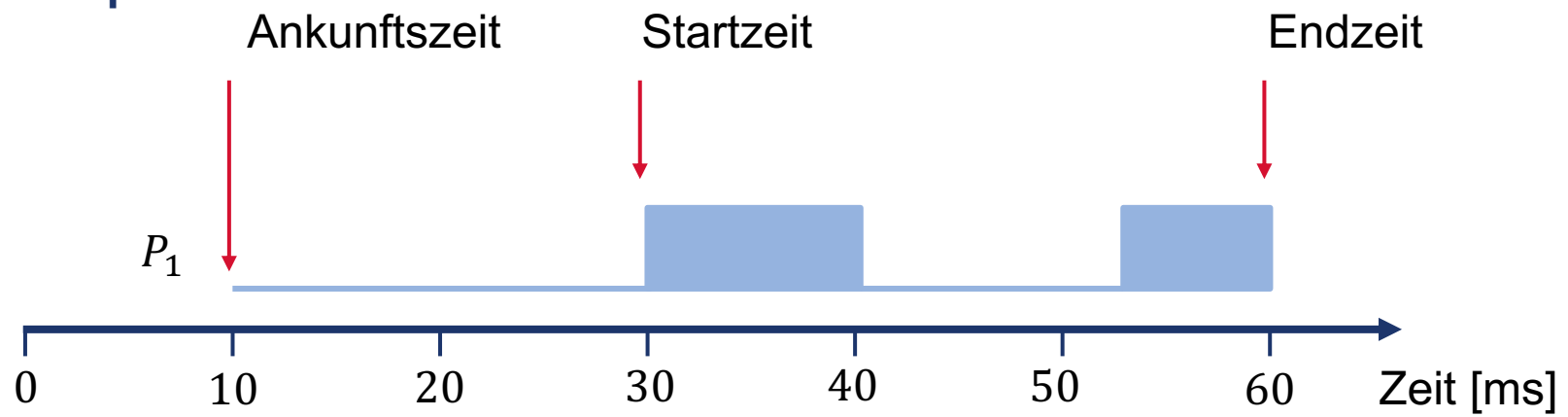
	P_1 [ms]
Ankunftszeit	10
Arbeitszeit	10 (17)
Startzeit	30
Wartezeit	20
Endzeit	

Metriken zur Prozessbeschreibung

- **Ankunftszeit** (arrival time): Zeitpunkt, an dem ein Prozess erzeugt wurde
- **CPU-Arbeitszeit** (burst time): Zeit, die der Prozess insgesamt rechnet

- **Startzeit** (starting time): Zeitpunkt, an dem ein Prozess zum ersten Mal die CPU nutzt
- **Wartezeit** (waiting time): Zeit, die ein Prozess in der Warteschlange verbringt
- **Endzeit** (finishing time): Zeitpunkt, an dem ein Prozess abgeschlossen ist

Beispiel



	P_1 [ms]
Ankunftszeit	10
Arbeitszeit	$10 + 7 = 17$
Startzeit	30
Wartezeit	33
Endzeit	

Bewertung von Scheduling-Algorithmen

- **Beendete Prozesse pro Zeiteinheit** (throughput)

  $$Durchsatz = \frac{Anzahl\ beendeter\ Prozesse}{Gesamtzeit}$$

- **Durchschnittliche Wartezeit** (avg. waiting time)

  $$AWT = \frac{\sum(Wartezeit)}{Anzahl\ der\ Prozesse}$$

- **Durchschnittliche Antwortzeit** (avg. response time)

  $$ART = \frac{\sum(Startzeit - Ankunftszeit)}{Anzahl\ der\ Prozesse}$$

- **Durchschnittliche Ausführungszeit** (avg. turnaround time)

  $$ATT = \frac{\sum(Endzeit - Ankunftszeit)}{Anzahl\ der\ Prozesse}$$

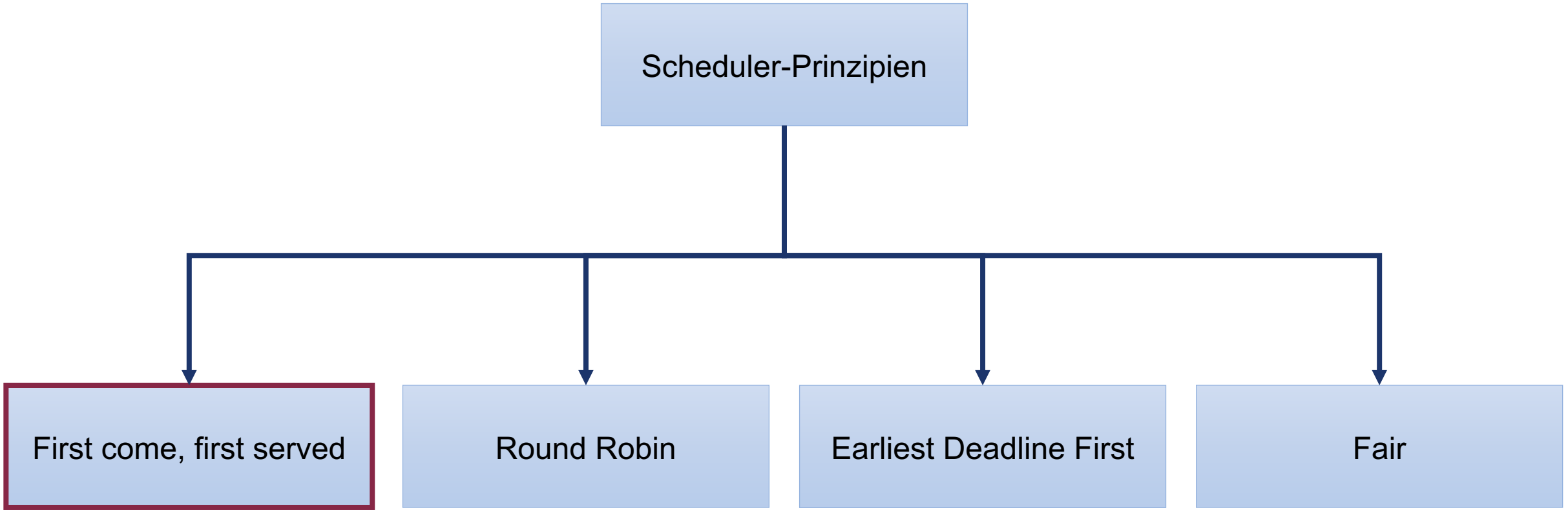
Annahmen für die folgenden Beispiele:

- $\mathcal{O}(1)$-Scheduler brauchen $1\ ms$ für die Prozessauswahl
- $\mathcal{O}(\log(n))$-Scheduler brauchen $1\ ms$ für die Prozessauswahl
- Wir vernachlässigen Wartezeiten für IO oder Speicher

Beispielprozesse:

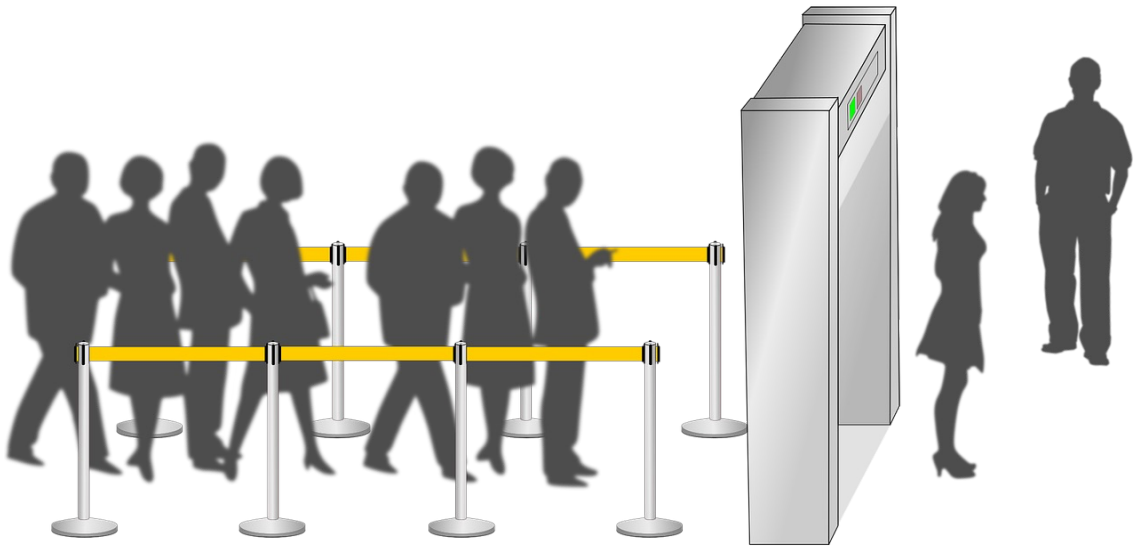
	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

Prinzipienübersicht



First Come First Served Scheduler (FCFS)

- Verhalten
  - Prozesse werden nach Startzeitpunkt sortiert
  - Prozesse werden nach FIFO-Prinzip gestartet
  - Aktive Prozesse werden nicht unterbrochen
  - Einfache Implementierung $\mathcal{O}(1)$



Beispiel: FCFS Scheduling

FCFS - Version 1 ($P_1 P_2 P_3 P_4$):

	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

FCFS - Version 1 ($P_1 P_2 P_3 P_4$):

	Startzeit [ms]	Wartezeit [ms]	Endzeit [ms]
P_1			
P_2			
P_3			
P_4			

Beispiel: FCFS Scheduling

	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

FCFS - Version 1 ($P_1P_2P_3P_4$):

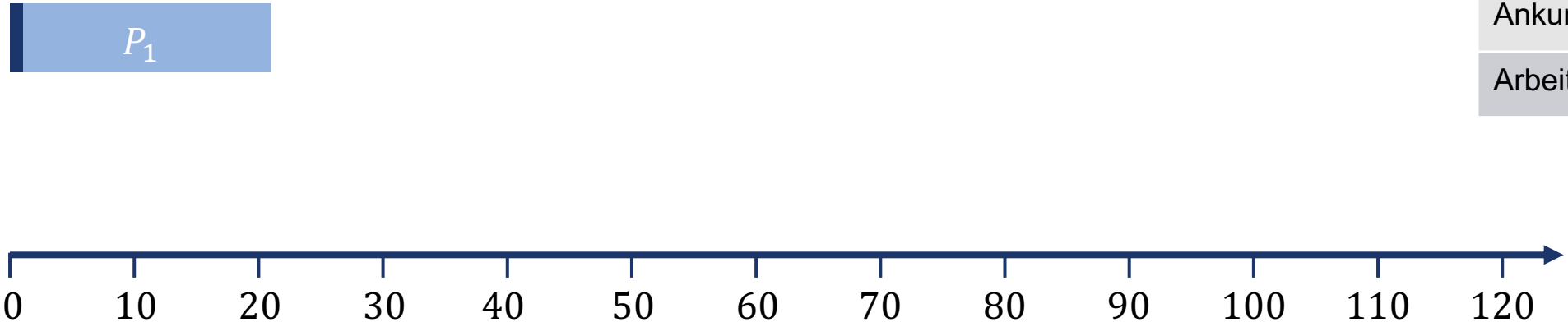
FCFS - Version 1 ($P_1P_2P_3P_4$):

	Startzeit [ms]	Wartezeit [ms]	Endzeit [ms]
P_1		1	
P_2		1	
P_3			
P_4			

Beispiel: FCFS Scheduling

	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

FCFS - Version 1 ($P_1P_2P_3P_4$):



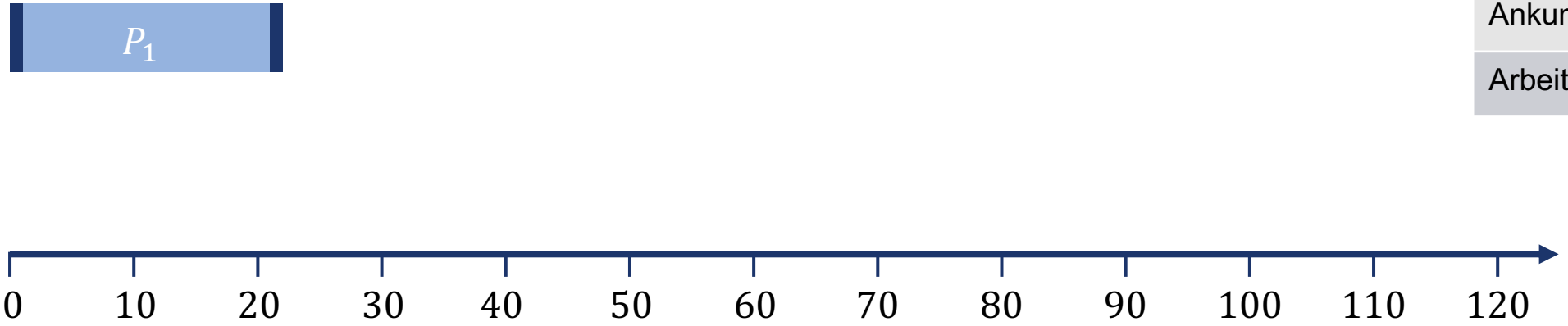
FCFS - Version 1 ($P_1P_2P_3P_4$):

	Startzeit [ms]	Wartezeit [ms]	Endzeit [ms]
P_1	1	1	$1 + 20 = 21$
P_2		$1 + 20 = 21$	
P_3		11	
P_4			

Beispiel: FCFS Scheduling

	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

FCFS - Version 1 ($P_1P_2P_3P_4$):



FCFS - Version 1 ($P_1P_2P_3P_4$):

	Startzeit [ms]	Wartezeit [ms]	Endzeit [ms]
P_1	1	1	21
P_2		$21 + 1 = 22$	
P_3		$11 + 1 = 12$	
P_4			

Beispiel: FCFS Scheduling

FCFS - Version 1 ($P_1 P_2 P_3 P_4$):

	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

FCFS - Version 1 ($P_1 P_2 P_3 P_4$):

	Startzeit [ms]	Wartezeit [ms]	Endzeit [ms]
P_1	1	1	21
P_2	22	22	$22 + 30 = 52$
P_3		$12 + 30 + 1 = 43$	
P_4		13	

Beispiel: FCFS Scheduling

FCFS - Version 1 ($P_1 P_2 P_3 P_4$):

	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

FCFS - Version 1 ($P_1 P_2 P_3 P_4$):

	Startzeit [ms]	Wartezeit [ms]	Endzeit [ms]
P_1	1	1	21
P_2	22	22	52
P_3	53	43	$53 + 40 = 93$
P_4		$13 + 40 + 1 = 54$	

Beispiel: FCFS Scheduling

	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

FCFS - Version 1 ($P_1P_2P_3P_4$):

FCFS - Version 1 ($P_1P_2P_3P_4$):

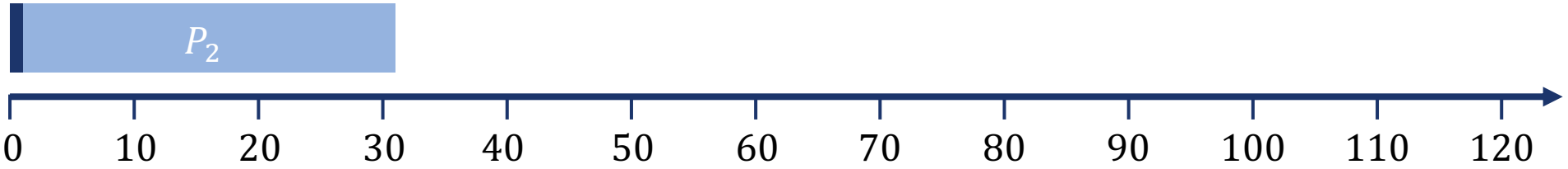
	Startzeit [ms]	Wartezeit [ms]	Endzeit [ms]
P_1	1	1	21
P_2	22	22	52
P_3	53	43	93
P_4	94	54	$94 + 10 = 104$

Beispiel: FCFS Scheduling

	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

FCFS - Version 1 ($P_1P_2P_3P_4$):

FCFS - Version 2 ($P_2P_1P_3P_4$):



FCFS - Version 1 ($P_1P_2P_3P_4$):

	Startzeit [ms]	Wartezeit [ms]	Endzeit [ms]
P_1	1	1	21
P_2	22	22	52
P_3	53	43	93
P_4	94	54	104

FCFS - Version 2 ($P_2P_1P_3P_4$):

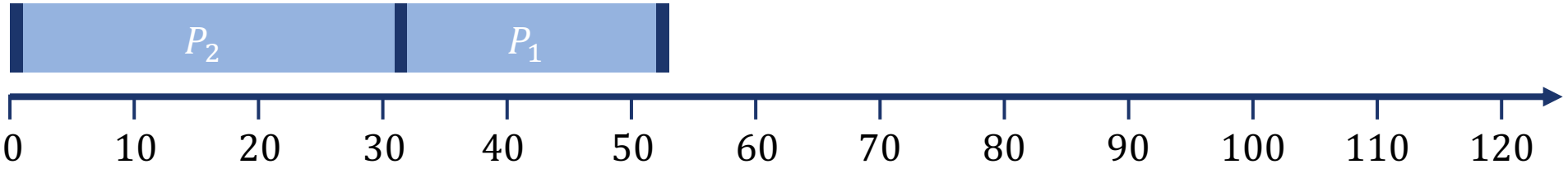
	Startzeit [ms]	Wartezeit [ms]	Endzeit [ms]
P_1		$1 + 30 = 31$	
P_2	1	1	31
P_3			
P_4			

Beispiel: FCFS Scheduling

	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

FCFS - Version 1 ($P_1P_2P_3P_4$):

FCFS - Version 2 ($P_2P_1P_3P_4$):



FCFS - Version 1 ($P_1P_2P_3P_4$):

	Startzeit [ms]	Wartezeit [ms]	Endzeit [ms]
P_1	1	1	21
P_2	22	22	52
P_3	53	43	93
P_4	94	54	104

FCFS - Version 2 ($P_2P_1P_3P_4$):

	Startzeit [ms]	Wartezeit [ms]	Endzeit [ms]
P_1	32	32	$32 + 20 = 52$
P_2	1	1	31
P_3		43	
P_4		13	

Beispiel: FCFS Scheduling

	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

FCFS - Version 1 ($P_1P_2P_3P_4$):

FCFS - Version 2 ($P_2P_1P_3P_4$):

FCFS - Version 1 ($P_1P_2P_3P_4$):

	Startzeit [ms]	Wartezeit [ms]	Endzeit [ms]
P_1	1	1	21
P_2	22	22	52
P_3	53	43	93
P_4	94	54	104

FCFS - Version 2 ($P_2P_1P_3P_4$):

	Startzeit [ms]	Wartezeit [ms]	Endzeit [ms]
P_1	32	32	52
P_2	1	1	31
P_3	53	43	93
P_4	94	54	104

Beispiel: FCFS Scheduling

Version 1 ($P_1 P_2 P_3 P_4$):

	Startzeit [ms]	Wartezeit [ms]	Endzeit [ms]
P_1	1	1	21
P_2	22	22	52
P_3	53	43	93
P_4	94	54	104

Version 2 ($P_2 P_1 P_3 P_4$):

	Startzeit [ms]	Wartezeit [ms]	Endzeit [ms]
P_1	32	32	52
P_2	1	1	31
P_3	53	43	93
P_4	94	54	104

Vergleich:

	Durchsatz	AWT	ART	ATT
Version 1				
Version 2				

	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

Beispiel: FCFS Scheduling

Version 1 ($P_1 P_2 P_3 P_4$):

	Startzeit [ms]	Wartezeit [ms]	Endzeit [ms]
P_1	1	1	21
P_2	22	22	52
P_3	53	43	93
P_4	94	54	104

Version 2 ($P_2 P_1 P_3 P_4$):

	Startzeit [ms]	Wartezeit [ms]	Endzeit [ms]
P_1	32	32	52
P_2	1	1	31
P_3	53	43	93
P_4	94	54	104

Vergleich:

	Durchsatz	AWT	ART	ATT
Version 1				
Version 2				

$$Durchsatz = \frac{Anzahl\ beendeter\ Prozesse}{Gesamtzeit}$$

	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

Beispiel: FCFS Scheduling

Version 1 ($P_1 P_2 P_3 P_4$):

	Startzeit [ms]	Wartezeit [ms]	Endzeit [ms]
P_1	1	1	21
P_2	22	22	52
P_3	53	43	93
P_4	94	54	104

Version 2 ($P_2 P_1 P_3 P_4$):

	Startzeit [ms]	Wartezeit [ms]	Endzeit [ms]
P_1	32	32	52
P_2	1	1	31
P_3	53	43	93
P_4	94	54	104

Vergleich:

	Durchsatz	AWT	ART	ATT
Version 1	$4/104$			
Version 2	$4/104$			

$$AWT = \frac{\sum(Wartezeit)}{Anzahl\ der\ Prozesse}$$

	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

Beispiel: FCFS Scheduling

Version 1 ($P_1 P_2 P_3 P_4$):

	Startzeit [ms]	Wartezeit [ms]	Endzeit [ms]
P_1	1	1	21
P_2	22	22	52
P_3	53	43	93
P_4	94	54	104

Version 2 ($P_2 P_1 P_3 P_4$):

	Startzeit [ms]	Wartezeit [ms]	Endzeit [ms]
P_1	32	32	52
P_2	1	1	31
P_3	53	43	93
P_4	94	54	104

Vergleich:

	Durchsatz	AWT	ART	ATT
Version 1	$4/104$	$30\ ms$		
Version 2	$4/104$	$32{,}5\ ms$		

$$ART = \frac{\sum(Startzeit\ -\ Ankunftszeit)}{Anzahl\ der\ Prozesse}$$

	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

Beispiel: FCFS Scheduling

Version 1 ($P_1 P_2 P_3 P_4$):

	Startzeit [ms]	Wartezeit [ms]	Endzeit [ms]
P_1	1	1	21
P_2	22	22	52
P_3	53	43	93
P_4	94	54	104

Version 2 ($P_2 P_1 P_3 P_4$):

	Startzeit [ms]	Wartezeit [ms]	Endzeit [ms]
P_1	32	32	52
P_2	1	1	31
P_3	53	43	93
P_4	94	54	104

Vergleich:

	Durchsatz	AWT	ART	ATT
Version 1	$4/104$	$30\ ms$	$30\ ms$	
Version 2	$4/104$	$32{,}5\ ms$	$32{,}5\ ms$	

$$ATT = \frac{\sum(Endzeit\ -\ Ankunftszeit)}{Anzahl\ der\ Prozesse}$$

	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

Beispiel: FCFS Scheduling

Version 1 ($P_1 P_2 P_3 P_4$):

	Startzeit [ms]	Wartezeit [ms]	Endzeit [ms]
P_1	1	1	21
P_2	22	22	52
P_3	53	43	93
P_4	94	54	104

Version 2 ($P_2 P_1 P_3 P_4$):

	Startzeit [ms]	Wartezeit [ms]	Endzeit [ms]
P_1	32	32	52
P_2	1	1	31
P_3	53	43	93
P_4	94	54	104

Vergleich:

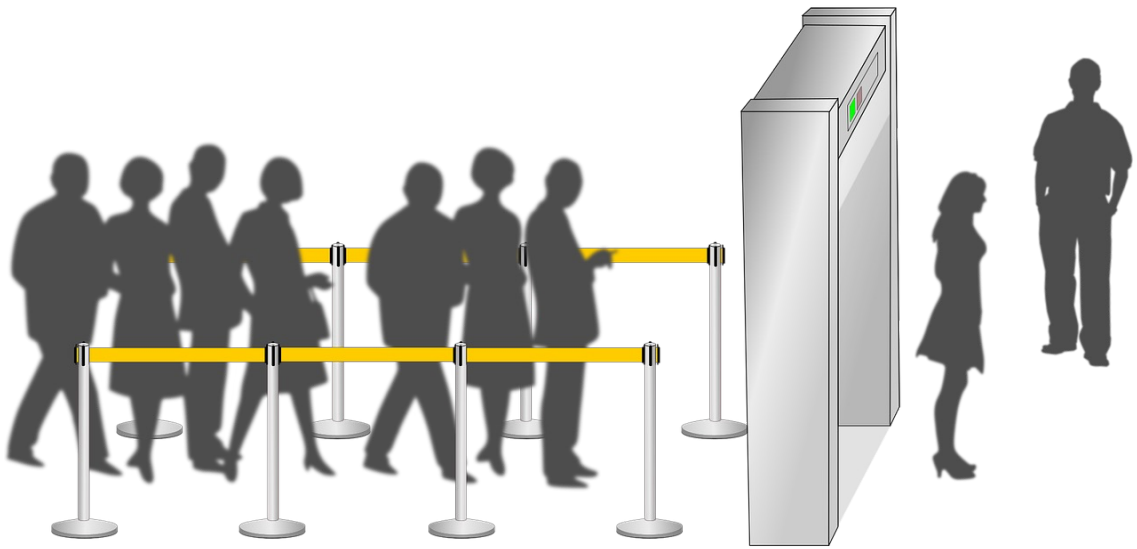
	Durchsatz	AWT	ART	ATT
Version 1	$4/104$	$30\ ms$	$30\ ms$	$55\ ms$
Version 2	$4/104$	$32{,}5\ ms$	$32{,}5\ ms$	$57{,}5\ ms$

Hinweis

Die Scheduling-Reihenfolge kann die wahrgenommene Performance bei interaktiven Systemen erheblich beeinflussen. Prinzipiell ist es von Vorteil, wenn zunächst kurzlebige Prozesse ausgeführt werden.

First Come First Served Scheduler (FCFS)

- Verhalten
  - Prozesse werden nach Startzeitpunkt sortiert
  - Prozesse werden nach FIFO-Prinzip gestartet
  - Aktiver Prozess wird nicht unterbrochen
  - Einfache Implementierung $\mathcal{O}(1)$
- Vorteile
  - Faires Scheduling-Verfahren
  - Sehr geringer Scheduling-Overhead
- Nachteile
  - Langlebige Prozesse führen zu langen Wartezeiten
  - Kurzlebige Prozesse können langsam wirken
- Einsatzgebiet
  - Eignet sich für Stapelverarbeitung

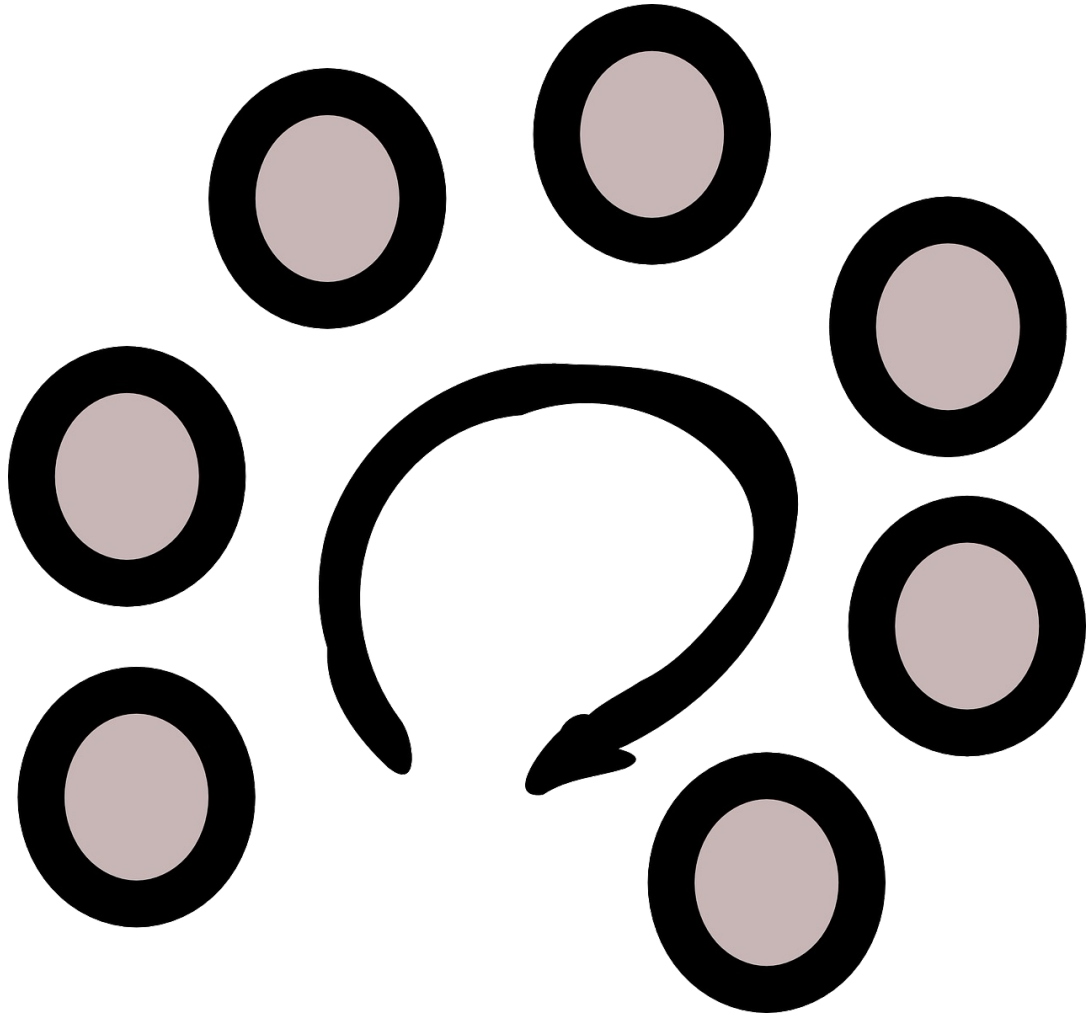


Zusammenfassung

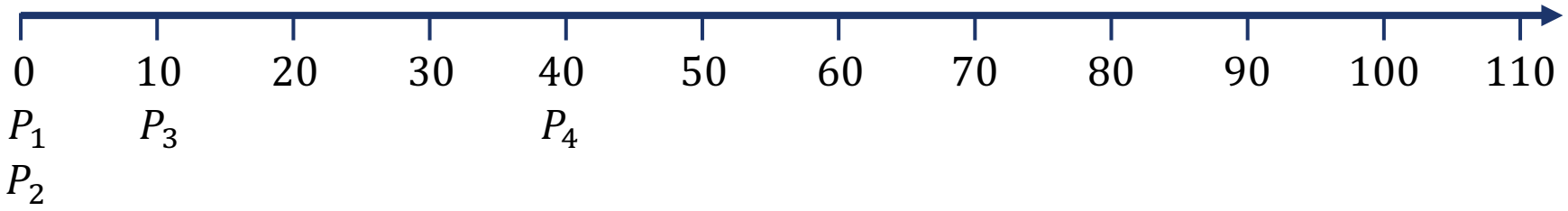
	FCFS	RR	EDF	FAIR
Nicht-Präemptiv				
Präemptiv				
Laufzeit				
Kooperatives Verhalten vorteilhaft				
Implementierungen				

Round-Robin Scheduler (RR)

- Verhalten
  - Scheduler verteilt feste Zeitscheiben der Länge d
  - Prozesse werden zyklisch aktiviert
    - Gleichbleibende Reihenfolge
    - Prozess darf komplette Zeitscheibe nutzen
    - Fertige Prozesse werden entfernt
    - Neue Prozesse kommen ans Ende
  - Einfache Implementierung $\mathcal{O}(1)$
  - Länge der Zeitscheiben bestimmen
    - Scheduling Overhead
    - Responsivität des Systems



Beispiel: Round-Robin Scheduling

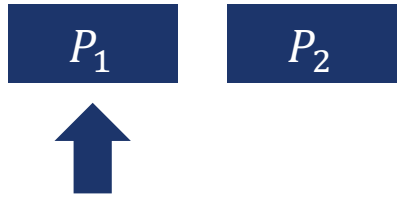


	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

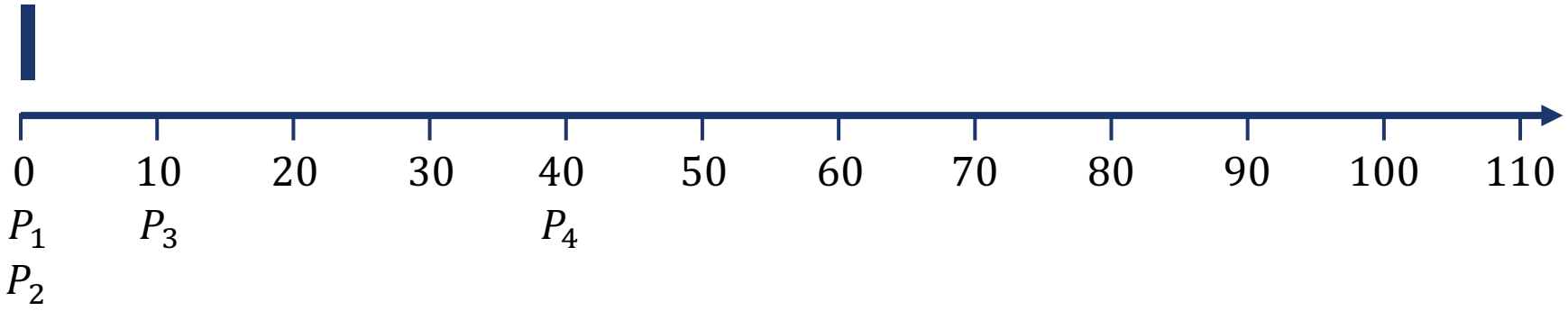
Annahme: $d = 10ms$

	Start [ms]	Warte [ms]	End [ms]
P_1			
P_2			
P_3			
P_4			

Nächster Prozess



Beispiel: Round-Robin Scheduling

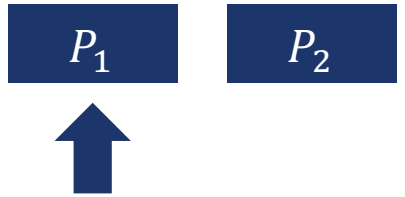


	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

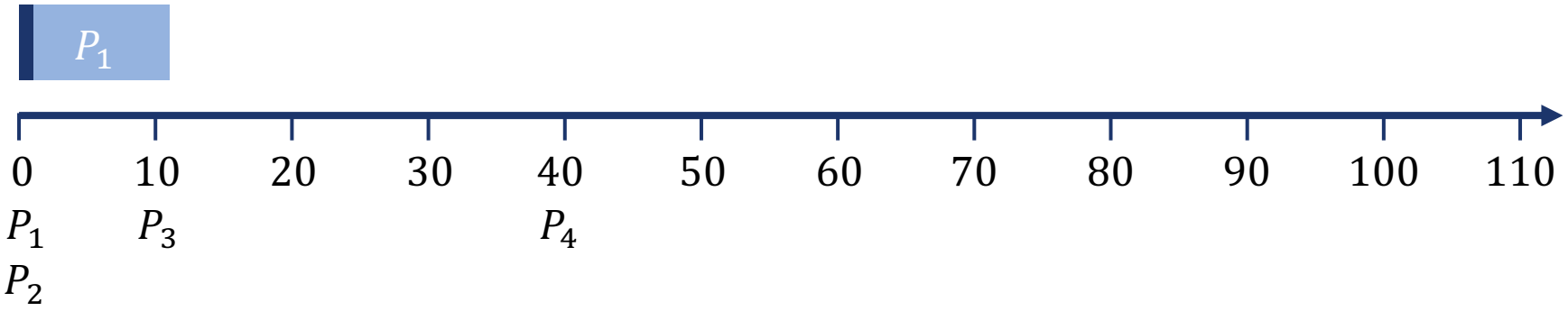
Annahme: $d = 10ms$

	Start [ms]	Warte [ms]	End [ms]
P_1		1	
P_2		1	
P_3			
P_4			

Nächster Prozess



Beispiel: Round-Robin Scheduling

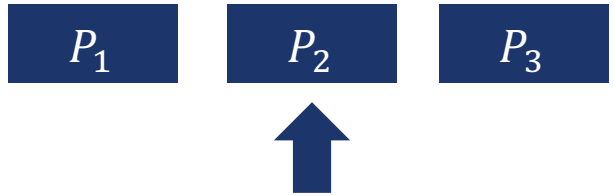


	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

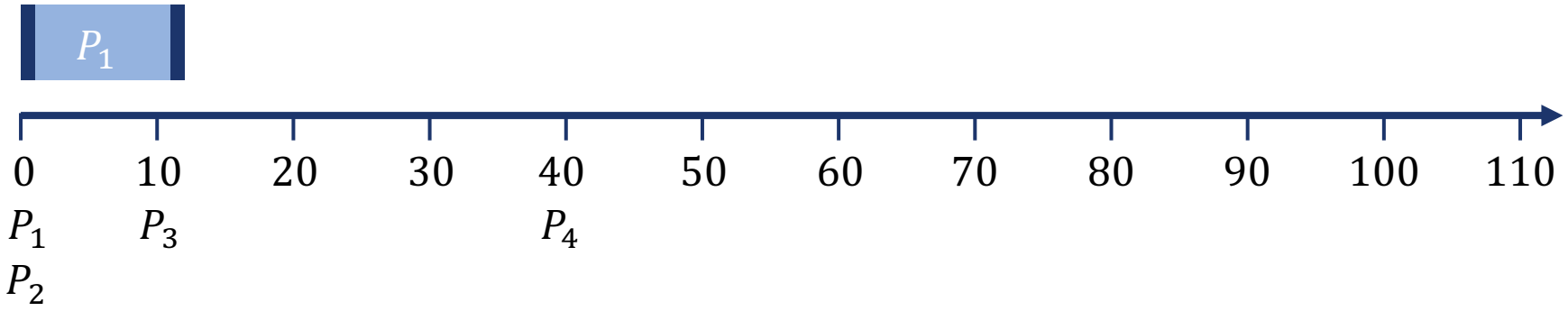
Annahme: $d = 10ms$

	Start [ms]	Warte [ms]	End [ms]
P_1	1	1	
P_2		11	
P_3		1	
P_4			

Nächster Prozess



Beispiel: Round-Robin Scheduling

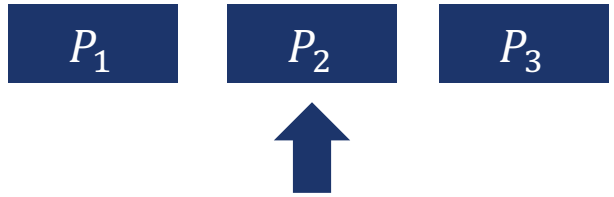


	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

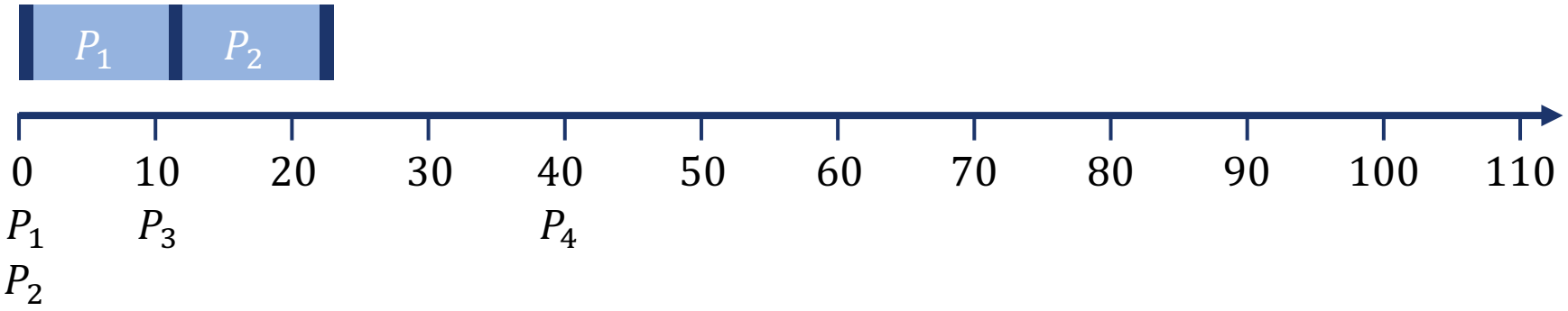
Annahme: $d = 10ms$

	Start [ms]	Warte [ms]	End [ms]
P_1	1	2	
P_2		12	
P_3		2	
P_4			

Nächster Prozess



Beispiel: Round-Robin Scheduling

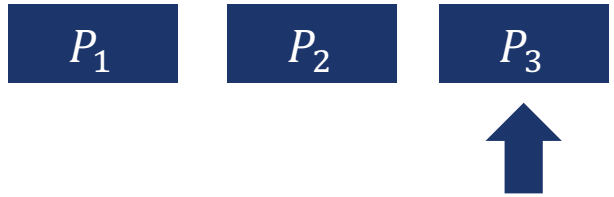


	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

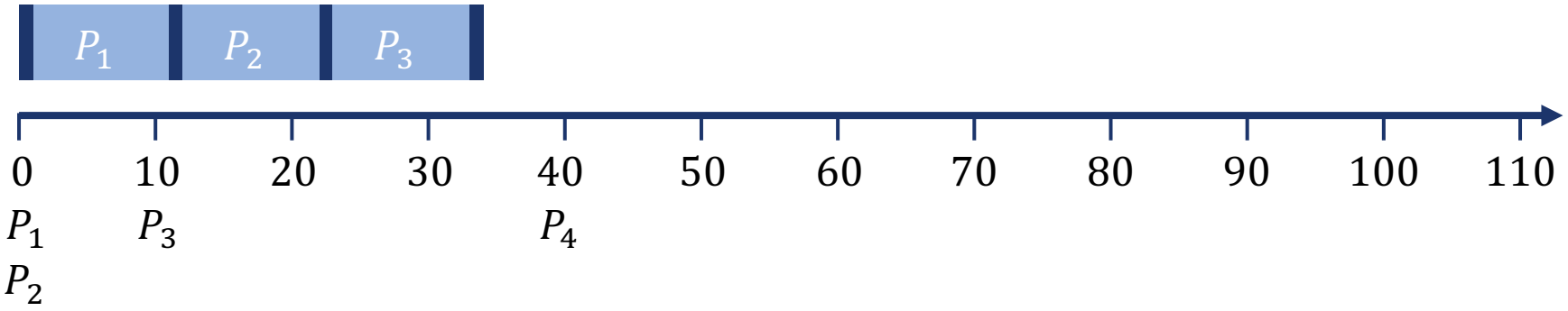
Annahme: $d = 10ms$

	Start [ms]	Warte [ms]	End [ms]
P_1	1	13	
P_2	12	13	
P_3		13	
P_4			

Nächster Prozess



Beispiel: Round-Robin Scheduling

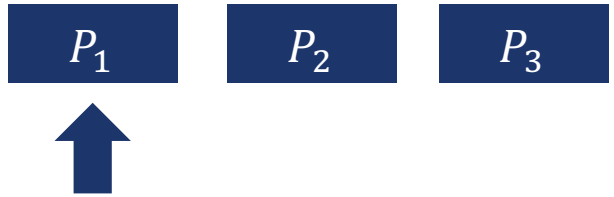


	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

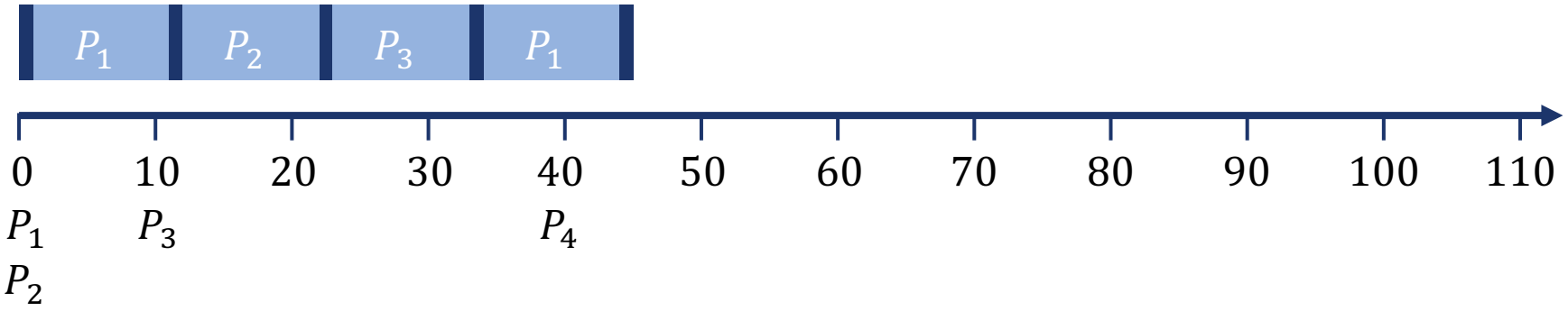
Annahme: $d = 10ms$

	Start [ms]	Warte [ms]	End [ms]
P_1	1	24	
P_2	12	24	
P_3	23	14	
P_4			

Nächster Prozess



Beispiel: Round-Robin Scheduling

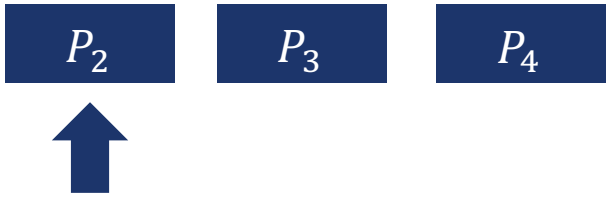


	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

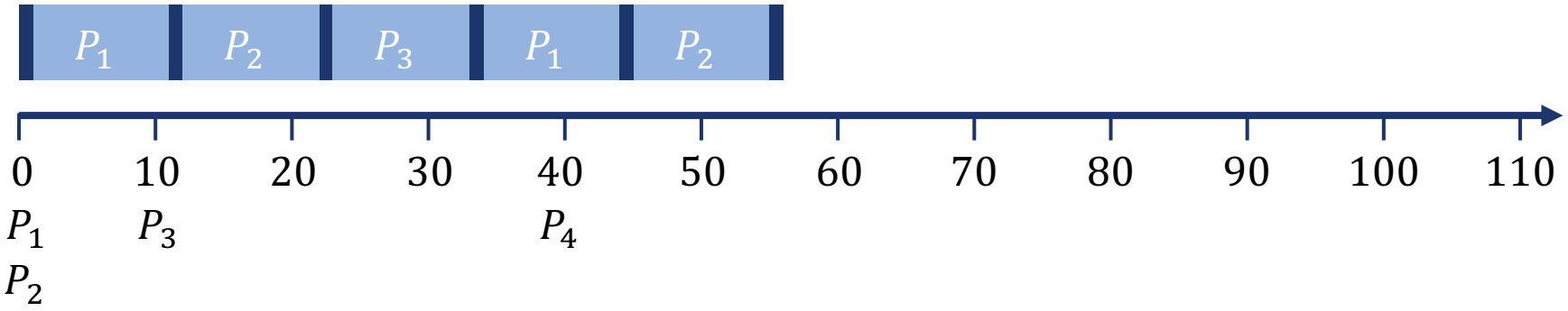
Annahme: $d = 10ms$

	Start [ms]	Warte [ms]	End [ms]
P_1	1	24	44
P_2	12	35	
P_3	23	25	
P_4		5	

Nächster Prozess



Beispiel: Round-Robin Scheduling

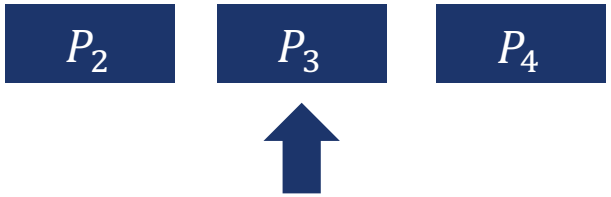


	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

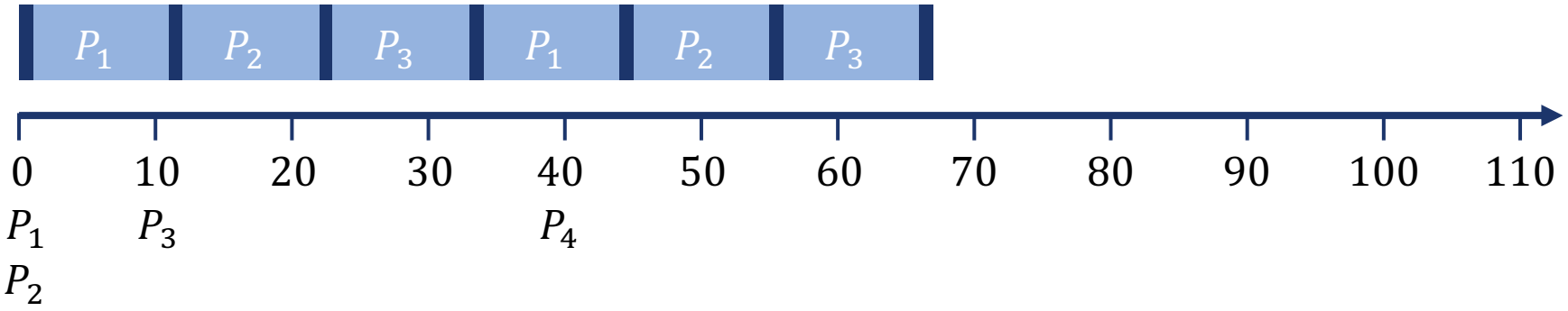
Annahme: $d = 10ms$

	Start [ms]	Warte [ms]	End [ms]
P_1	1	24	44
P_2	12	36	
P_3	23	36	
P_4		16	

Nächster Prozess



Beispiel: Round-Robin Scheduling

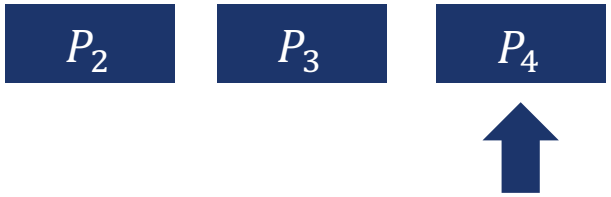


	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

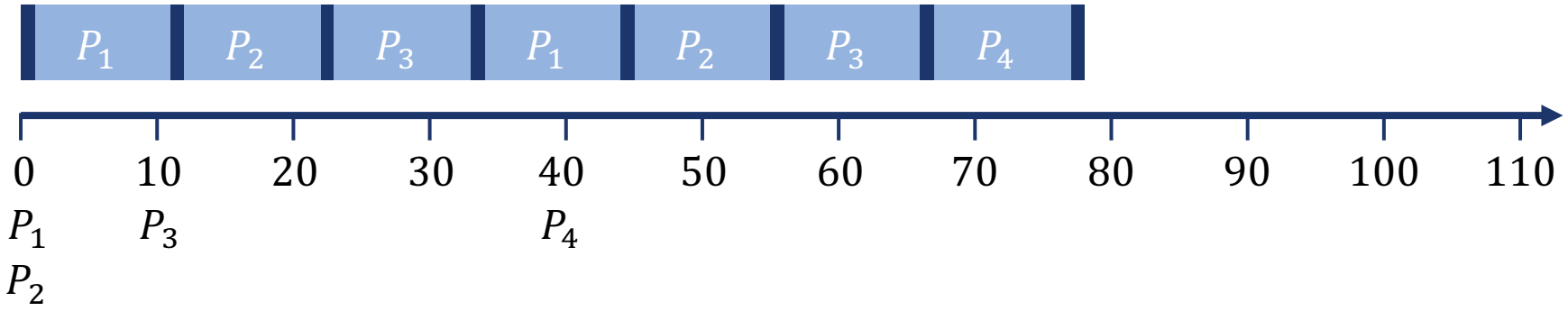
Annahme: $d = 10ms$

	Start [ms]	Warte [ms]	End [ms]
P_1	1	24	44
P_2	12	47	
P_3	23	37	
P_4		27	

Nächster Prozess



Beispiel: Round-Robin Scheduling

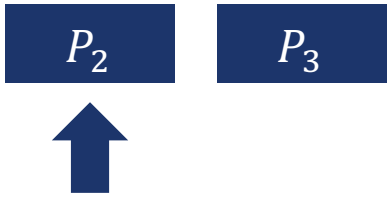


	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

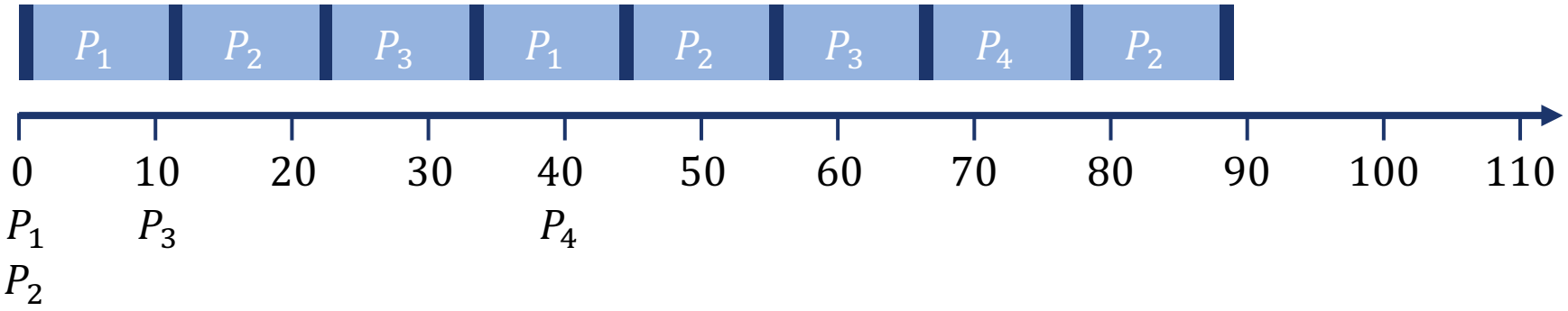
Annahme: $d = 10ms$

	Start [ms]	Warte [ms]	End [ms]
P_1	1	24	44
P_2	12	58	
P_3	23	48	
P_4	67	27	77

Nächster Prozess



Beispiel: Round-Robin Scheduling

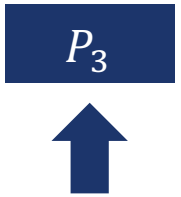


	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

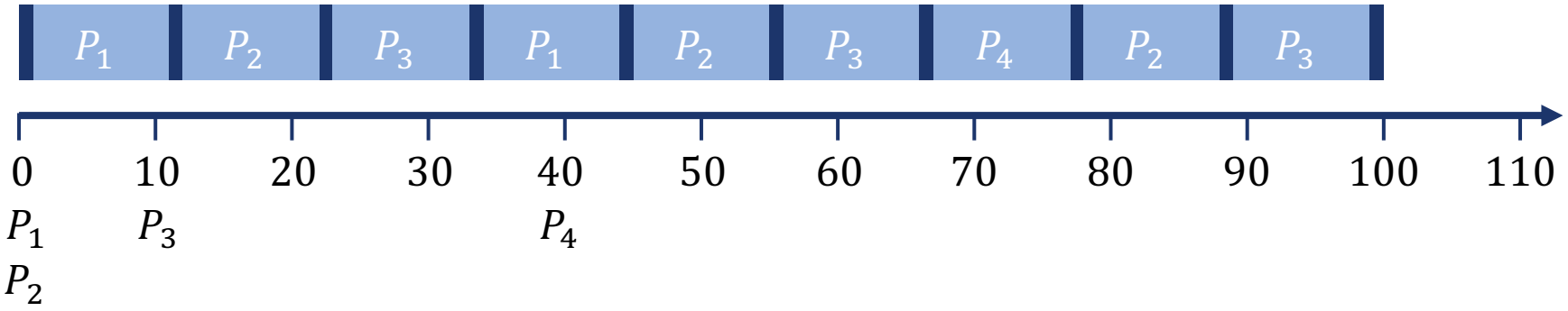
Annahme: $d = 10ms$

	Start [ms]	Warte [ms]	End [ms]
P_1	1	24	44
P_2	12	58	88
P_3	23	59	
P_4	67	27	77

Nächster Prozess



Beispiel: Round-Robin Scheduling

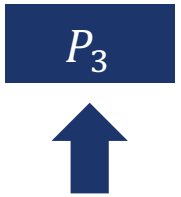


	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

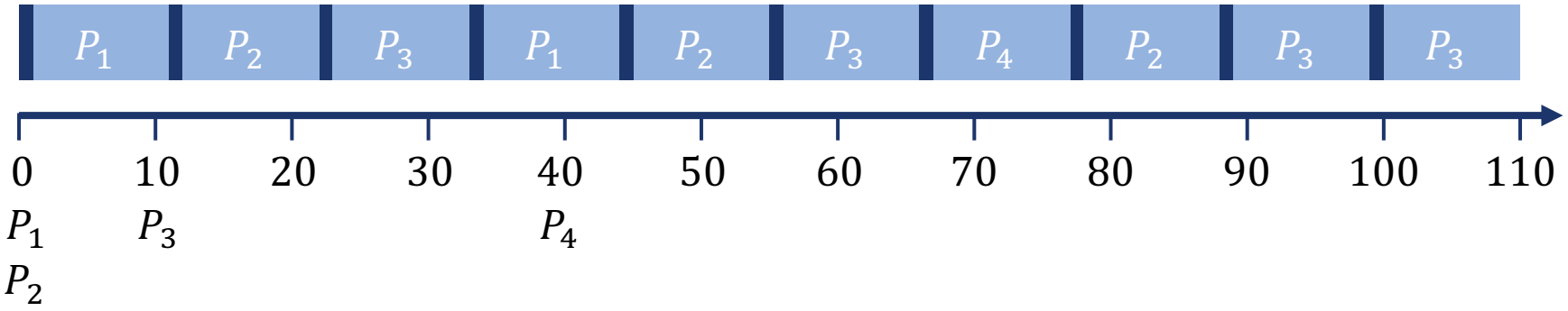
Annahme: $d = 10ms$

	Start [ms]	Warte [ms]	End [ms]
P_1	1	24	44
P_2	12	58	88
P_3	23	60	
P_4	67	27	77

Nächster Prozess



Beispiel: Round-Robin Scheduling

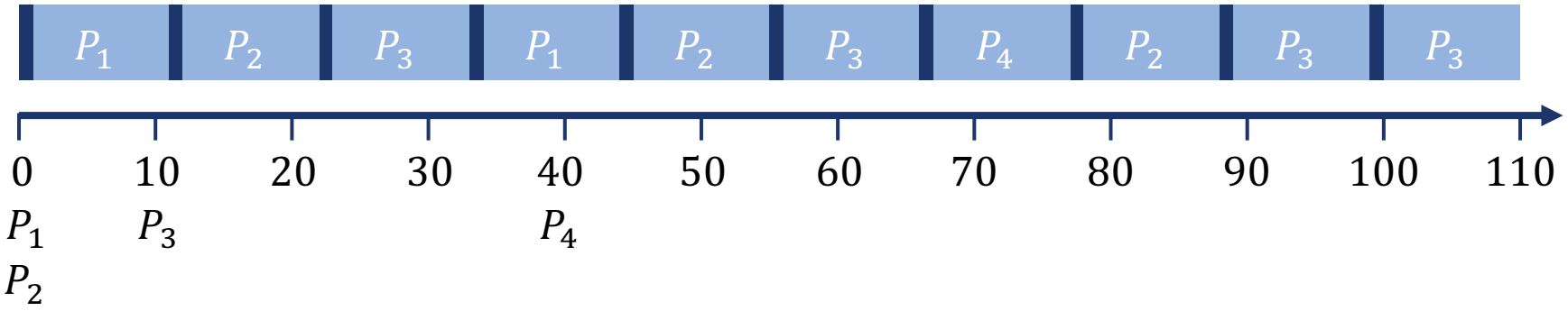


	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

Annahme: $d = 10ms$

	Start [ms]	Warte [ms]	End [ms]
P_1	1	24	44
P_2	12	58	88
P_3	23	60	110
P_4	67	27	77

Beispiel: Round-Robin Scheduling



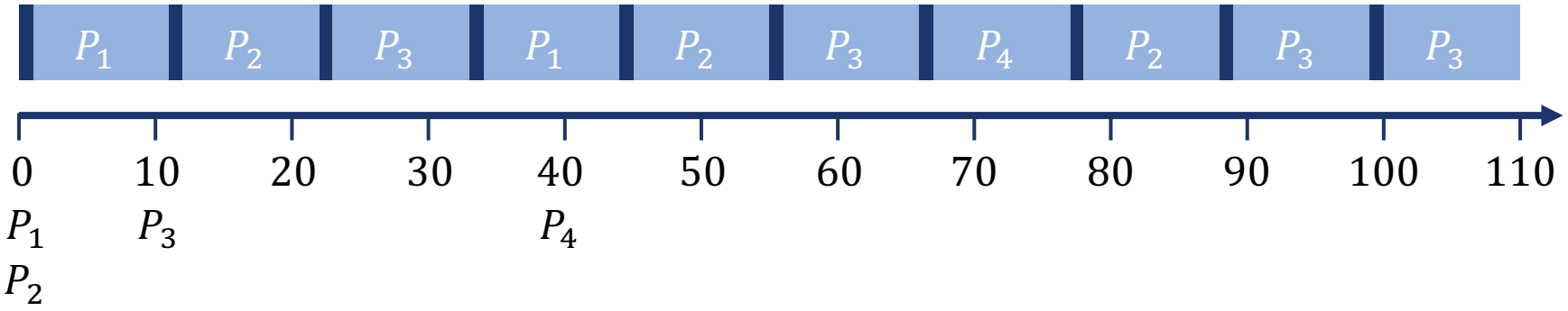
	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

Annahme: $d = 10ms$

	Start [ms]	Warte [ms]	End [ms]
P_1	1	24	44
P_2	12	58	88
P_3	23	60	110
P_4	67	27	77

	Durchsatz	AWT	ART	ATT
FCFS - 1	4/104	30,00 ms	30,00 ms	55,00 ms
FCFS - 2	4/104	32,50 ms	32,50 ms	57,50 ms
RR				

Beispiel: Round-Robin Scheduling



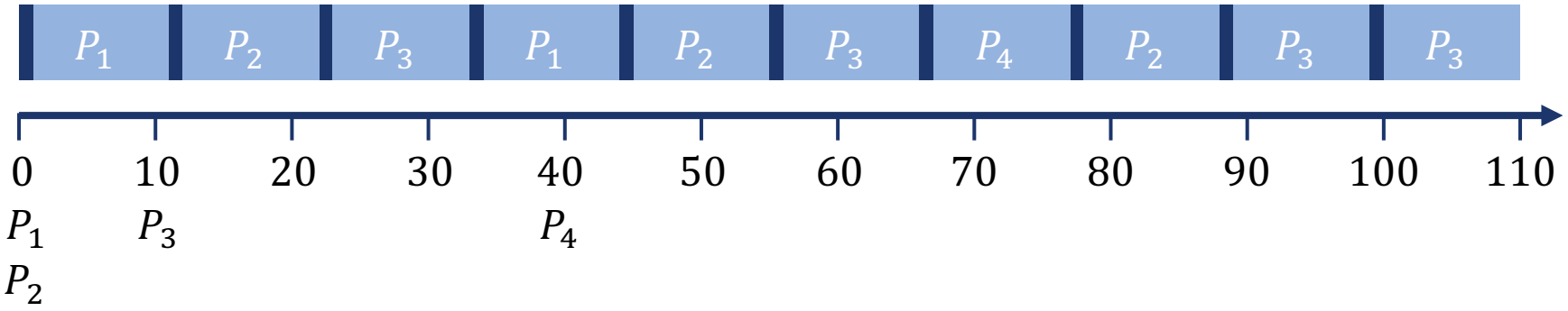
	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

Annahme: $d = 10ms$

	Start [ms]	Warte [ms]	End [ms]
P_1	1	24	44
P_2	12	58	88
P_3	23	60	110
P_4	67	27	77

	Durchsatz	AWT	ART	ATT
FCFS - 1	4/104	30,00 ms	30,00 ms	55,00 ms
FCFS - 2	4/104	32,50 ms	32,50 ms	57,50 ms
RR	4/110			

Beispiel: Round-Robin Scheduling



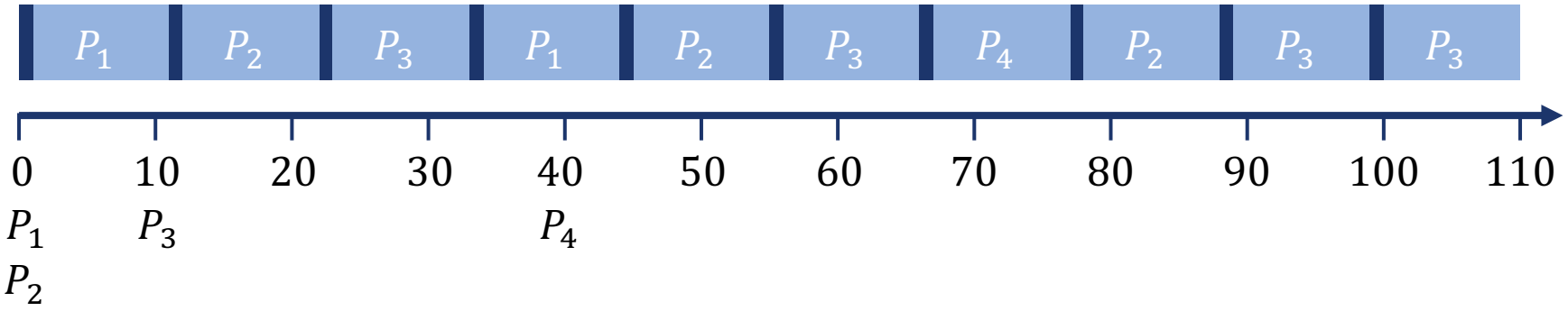
	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

Annahme: $d = 10ms$

	Start [ms]	Warte [ms]	End [ms]
P_1	1	24	44
P_2	12	58	88
P_3	23	60	110
P_4	67	27	77

	Durchsatz	AWT	ART	ATT
FCFS - 1	4/104	30,00 ms	30,00 ms	55,00 ms
FCFS - 2	4/104	32,50 ms	32,50 ms	57,50 ms
RR	4/110			

Beispiel: Round-Robin Scheduling



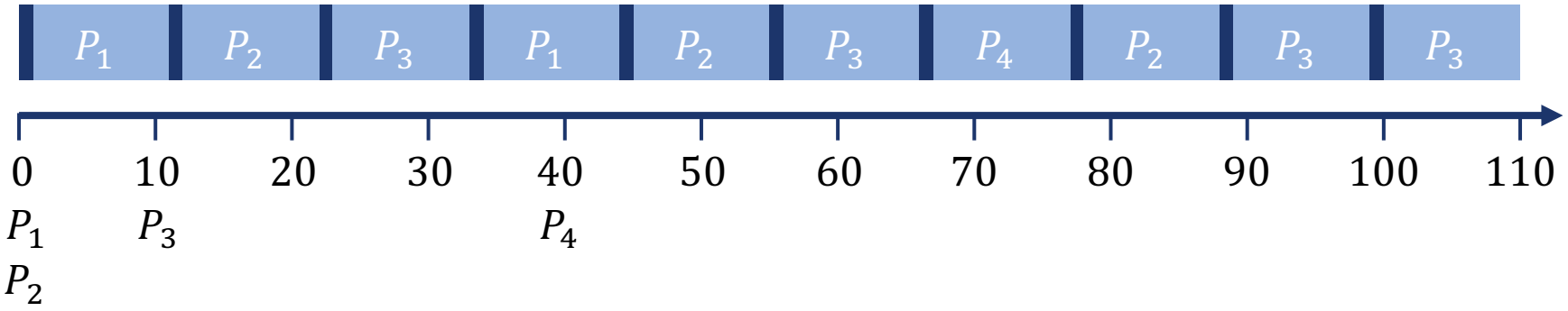
	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

Annahme: $d = 10ms$

	Start [ms]	Warte [ms]	End [ms]
P_1	1	24	44
P_2	12	58	88
P_3	23	60	110
P_4	67	27	77

	Durchsatz	AWT	ART	ATT
FCFS - 1	4/104	30,00 ms	30,00 ms	55,00 ms
FCFS - 2	4/104	32,50 ms	32,50 ms	57,50 ms
RR	4/110	42,25 ms		

Beispiel: Round-Robin Scheduling



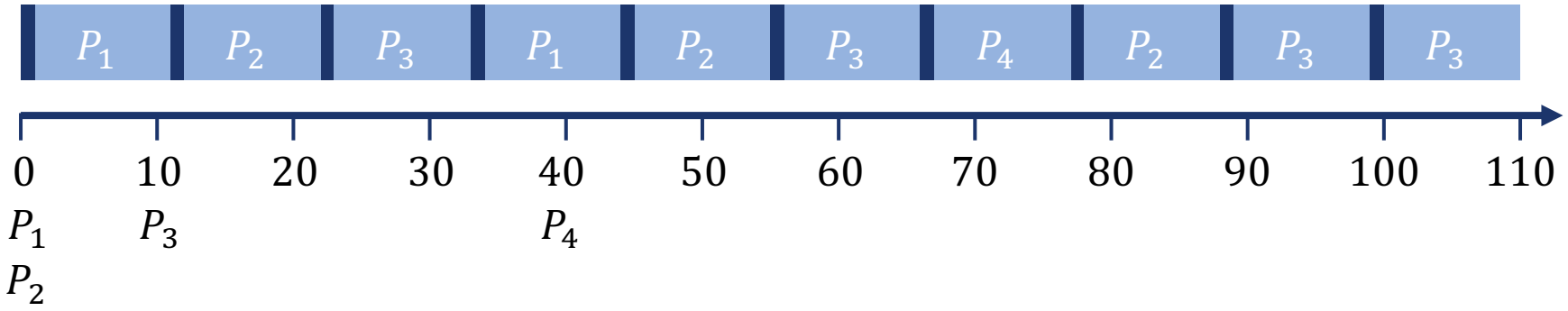
	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

Annahme: $d = 10ms$

	Start [ms]	Warte [ms]	End [ms]
P_1	1	24	44
P_2	12	58	88
P_3	23	60	110
P_4	67	27	77

	Durchsatz	AWT	ART	ATT
FCFS - 1	4/104	30,00 ms	30,00 ms	55,00 ms
FCFS - 2	4/104	32,50 ms	32,50 ms	57,50 ms
RR	4/110	42,25 ms	13,25 ms	

Beispiel: Round-Robin Scheduling



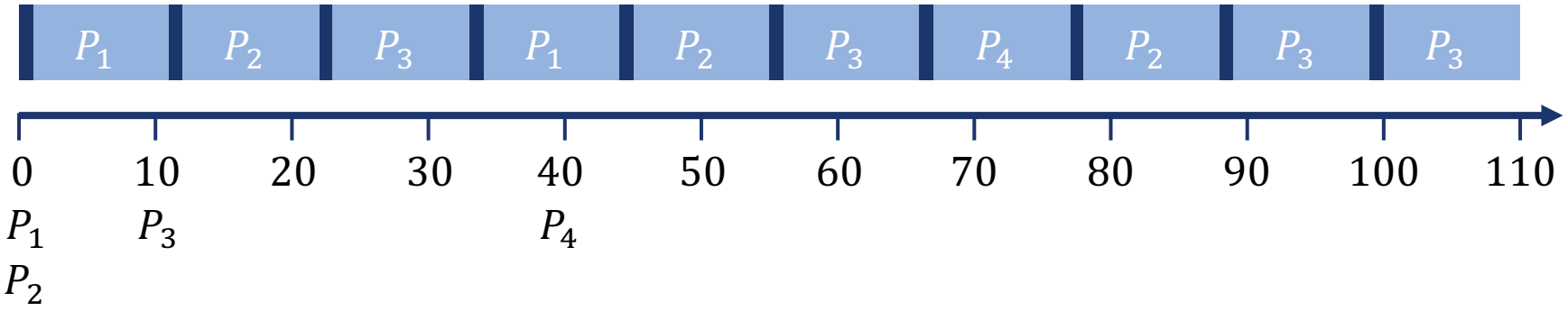
	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

Annahme: $d = 10ms$

	Start [ms]	Warte [ms]	End [ms]
P_1	1	24	44
P_2	12	58	88
P_3	23	60	110
P_4	67	27	77

	Durchsatz	AWT	ART	ATT
FCFS - 1	4/104	30,00 ms	30,00 ms	55,00 ms
FCFS - 2	4/104	32,50 ms	32,50 ms	57,50 ms
RR	4/110	42,25 ms	13,25 ms	67,25 ms

Beispiel: Round-Robin Scheduling



	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10

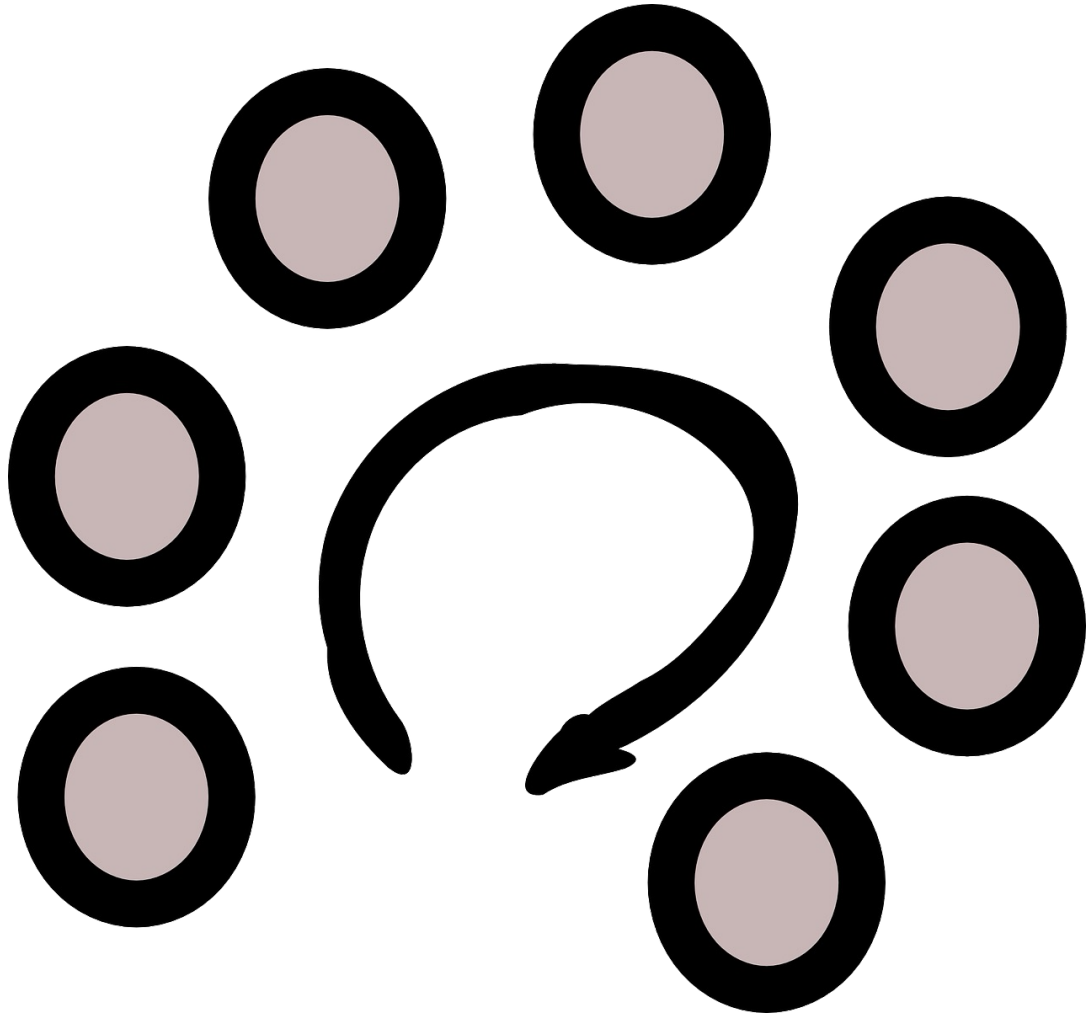
Annahme: $d = 10ms$

	Start [ms]	Warte [ms]	End [ms]
P_1	1	24	44
P_2	12	58	88
P_3	23	60	110
P_4	67	27	77

	Durchsatz	AWT	ART	ATT
FCFS - 1	4/104	30,00 ms	30,00 ms	55,00 ms
FCFS - 2	4/104	32,50 ms	32,50 ms	57,50 ms
RR	4/110	42,25 ms	13,25 ms	67,25 ms

Round-Robin Scheduler (RR)

- Verhalten
  - Scheduler verteilt feste Zeitscheiben der Länge d
  - Prozesse werden zyklisch aktiviert
    - Gleichbleibende Reihenfolge
    - Prozess darf komplette Zeitscheibe nutzen
    - Fertige Prozesse werden entfernt
    - Neue Prozesse kommen ans Ende
  - Einfache Implementierung $\mathcal{O}(1)$
  - Länge der Zeitscheiben bestimmen
    - Scheduling Overhead
    - Responsivität des Systems
- Vorteile
  - Faire CPU-Verteilung
- Nachteile
  - Overhead durch regelmäßige Prozesswechsel
  - Kooperatives Verhalten hat keinen Vorteil



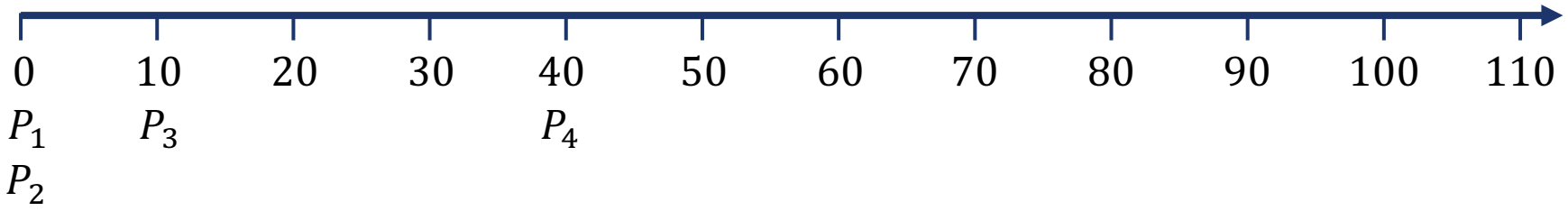
Zusammenfassung

	FCFS	RR	EDF	FAIR
Nicht-Präemptiv				
Präemptiv				
Laufzeit				
Kooperatives Verhalten vorteilhaft				
Implementierungen				

Earliest Deadline First

- Verhalten
  - Prozesse haben feste Endzeit
    - Sortierung nach Endzeit
    - Nächste Endzeit wird als erstes gewählt
  - Neuberechnung notwendig
    - Bei Aktivierung eines Prozesses
    - Terminierung des aktiven Prozesses
  - Sowohl präemptiv als auch nicht-präemptiv

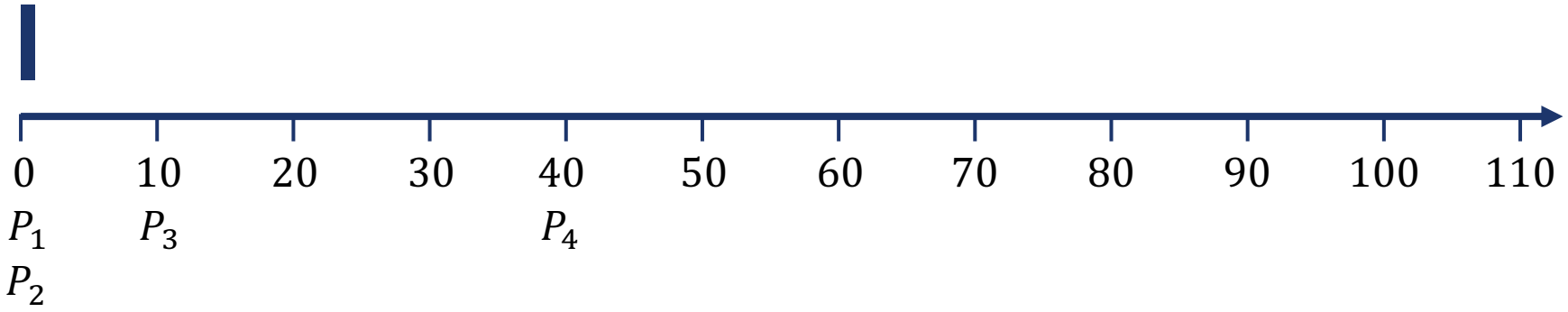
Beispiel: EDF Scheduling



	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10
Deadline	35	120	80	60

	Start [ms]	Warte [ms]	End [ms]
P_1			
P_2			
P_3			
P_4			

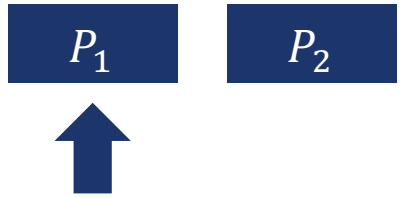
Beispiel: EDF Scheduling



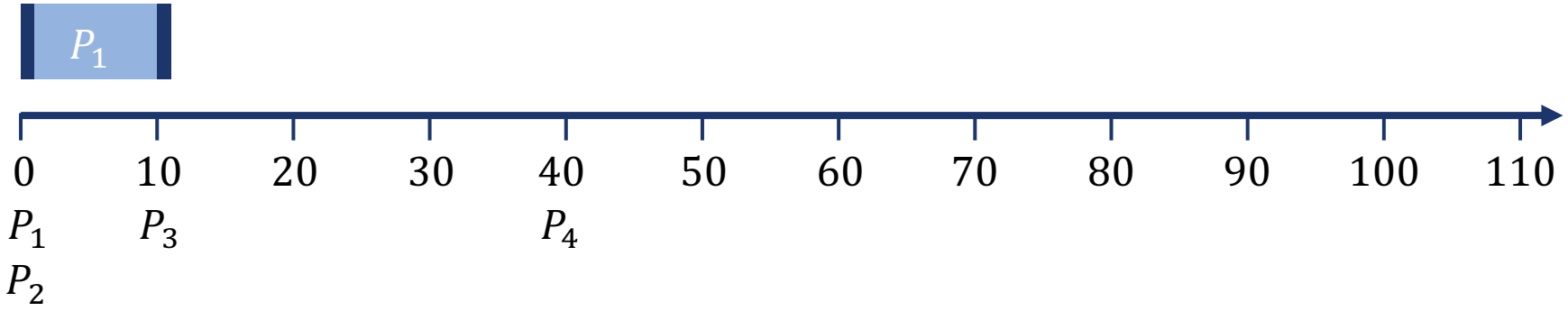
	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10
Deadline	35	120	80	60

	Start [ms]	Warte [ms]	End [ms]
P_1	1	1	
P_2		1	
P_3			
P_4			

Nächster Prozess



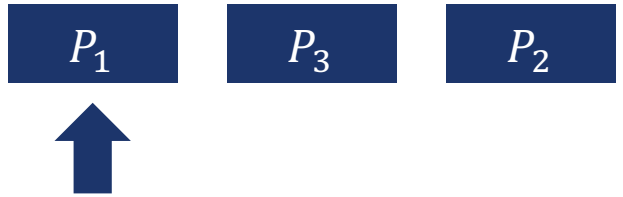
Beispiel: EDF Scheduling



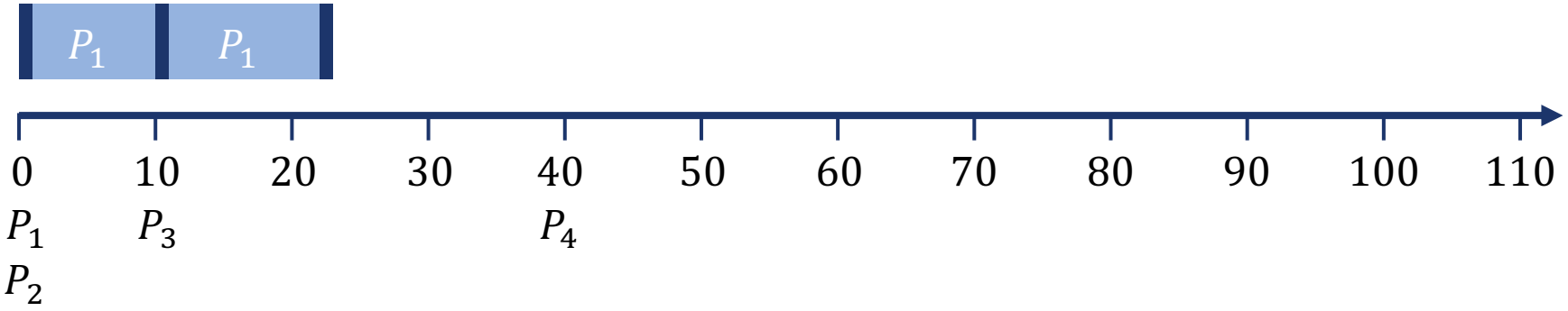
	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10
Deadline	35	120	80	60

	Start [ms]	Warte [ms]	End [ms]
P_1	1	2	
P_2		12	
P_3		1	
P_4			

Nächster Prozess



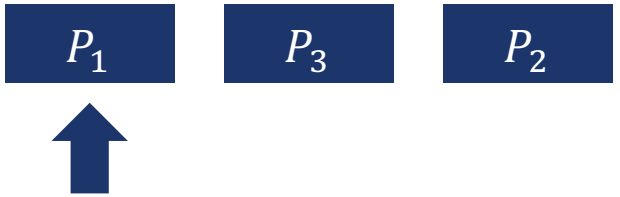
Beispiel: EDF Scheduling



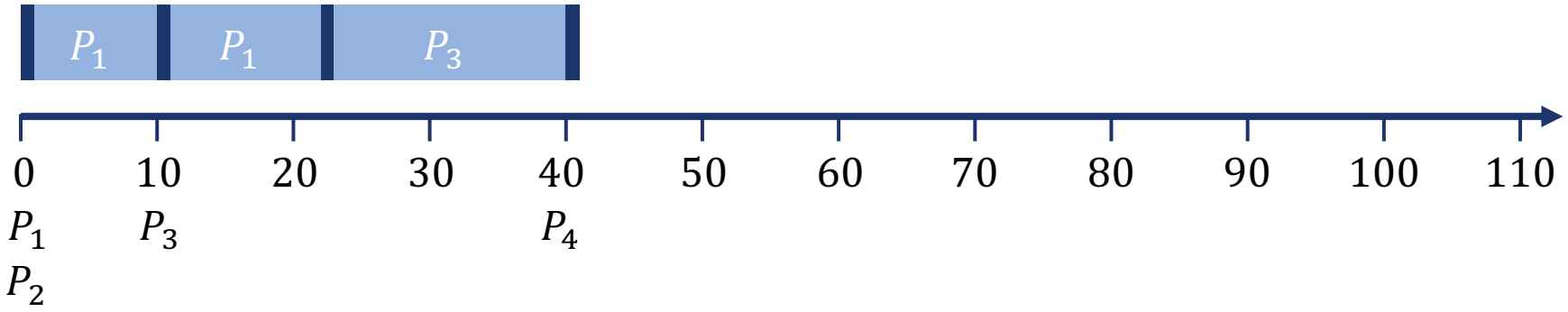
	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10
Deadline	35	120	80	60

	Start [ms]	Warte [ms]	End [ms]
P_1	1	2	22
P_2		23	
P_3		12	
P_4			

Nächster Prozess



Beispiel: EDF Scheduling

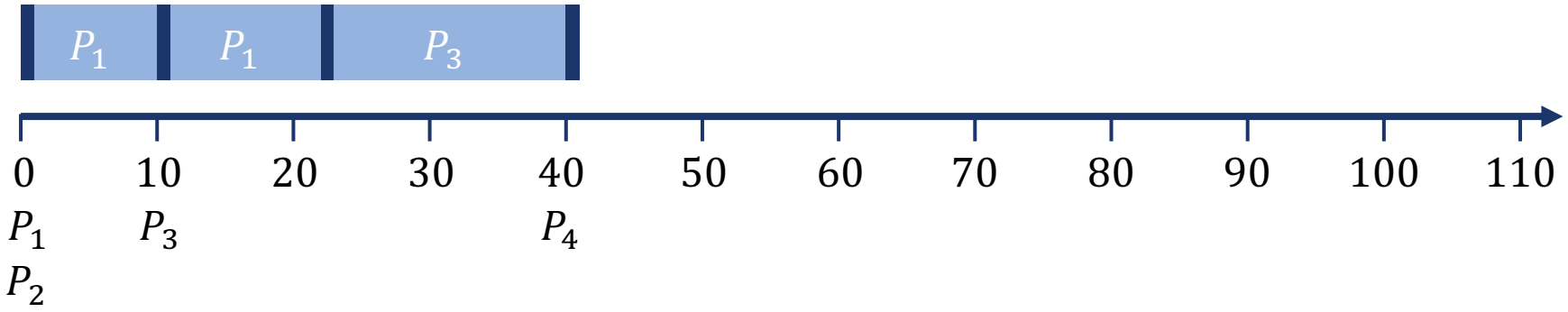


	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10
Deadline	35	120	80	60

	Start [ms]	Warte [ms]	End [ms]
P_1	1	2	22
P_2		41	
P_3	23	13	
P_4		1	

Nächster Prozess

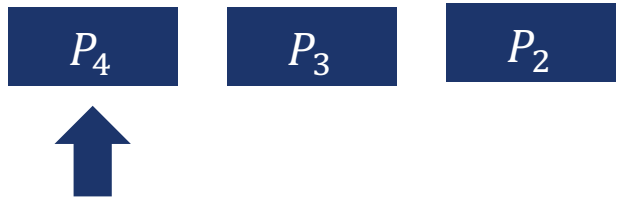
Beispiel: EDF Scheduling



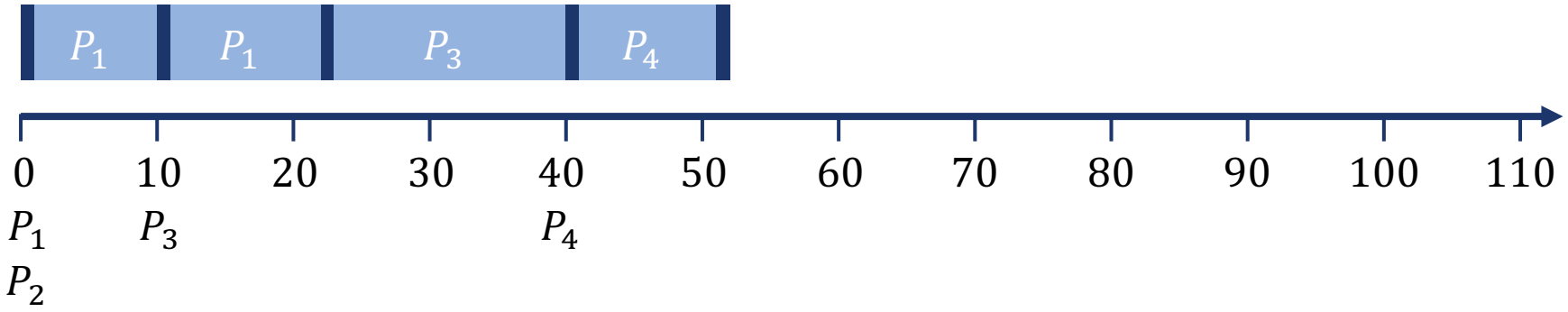
	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10
Deadline	35	120	80	60

	Start [ms]	Warte [ms]	End [ms]
P_1	1	2	22
P_2		41	
P_3	23	13	
P_4		1	

Nächster Prozess



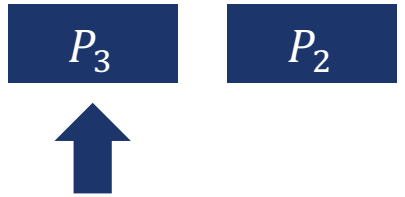
Beispiel: EDF Scheduling



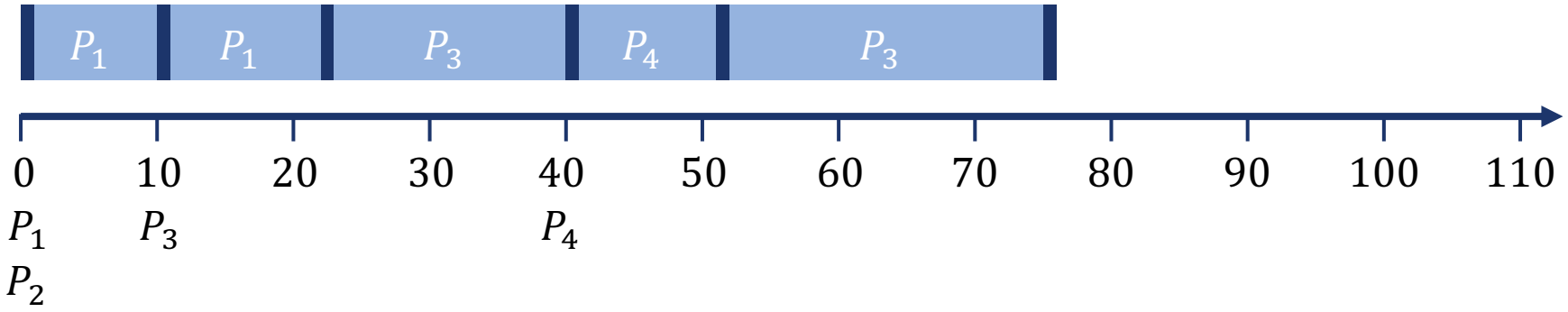
	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10
Deadline	35	120	80	60

	Start [ms]	Warte [ms]	End [ms]
P_1	1	2	22
P_2		52	
P_3	23	24	
P_4	41	1	51

Nächster Prozess



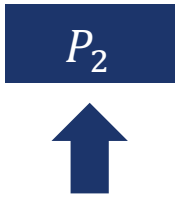
Beispiel: EDF Scheduling



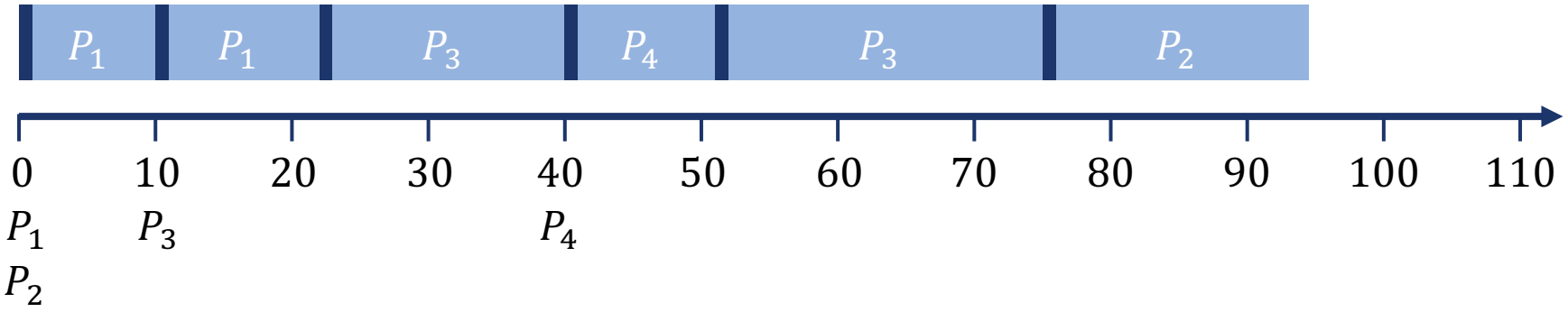
	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10
Deadline	35	120	80	60

	Start [ms]	Warte [ms]	End [ms]
P_1	1	2	22
P_2		76	
P_3	23	24	75
P_4	41	1	51

Nächster Prozess



Beispiel: EDF Scheduling

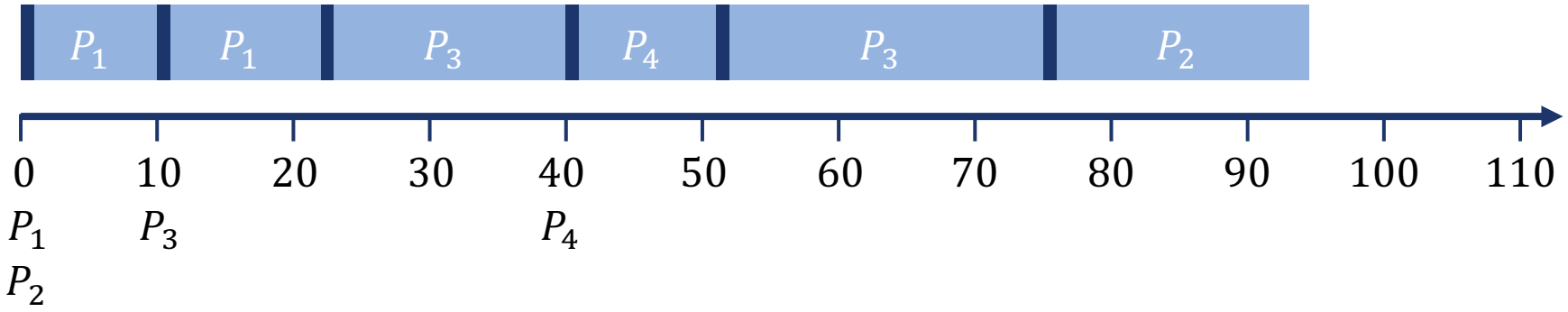


	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10
Deadline	35	120	80	60

	Start [ms]	Warte [ms]	End [ms]
P_1	1	2	22
P_2	76	76	96
P_3	23	24	75
P_4	41	1	51

Nächster Prozess

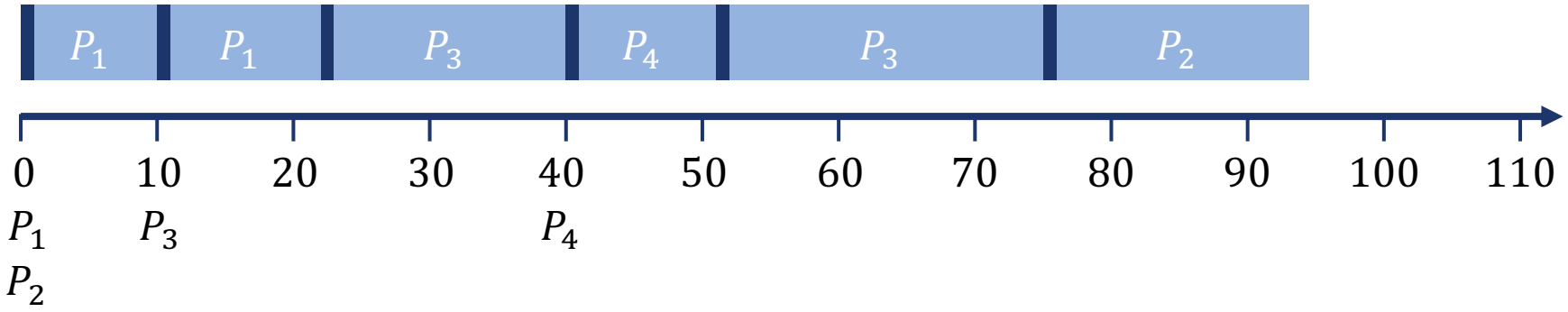
Beispiel: EDF Scheduling



	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10
Deadline	35	120	80	60

	Start [ms]	Warte [ms]	End [ms]
P_1	1	2	22
P_2	76	76	96
P_3	23	24	75
P_4	41	1	51

Beispiel: EDF Scheduling

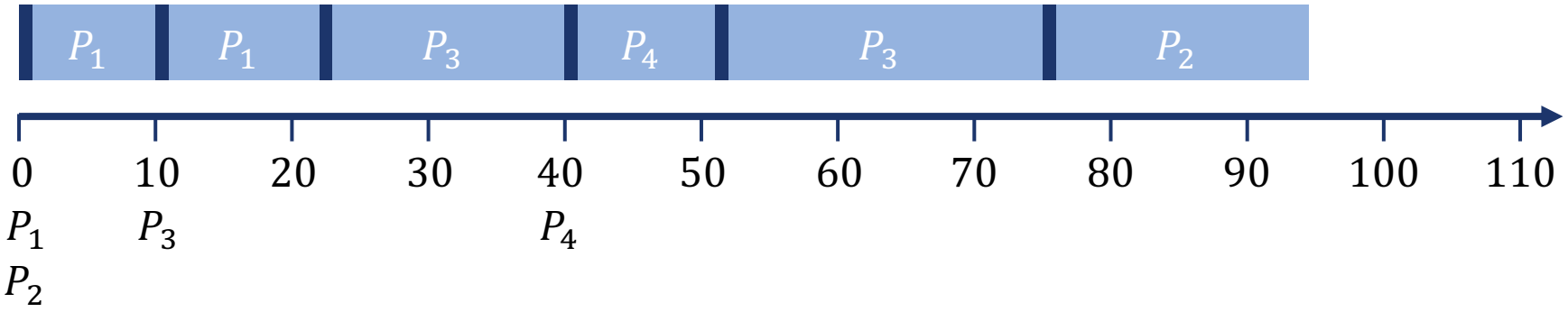


	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10
Deadline	35	120	80	60

	Start [ms]	Warte [ms]	End [ms]
P_1	1	2	22
P_2	76	76	96
P_3	23	24	75
P_4	41	1	51

	Durchsatz	AWT	ART	ATT
FCFS - 1	4/104	30,00 ms	30,00 ms	55,00 ms
FCFS - 2	4/104	32,50 ms	32,50 ms	57,50 ms
RR	4/110	42,25 ms	13,25 ms	67,25 ms
EDF				

Beispiel: EDF Scheduling

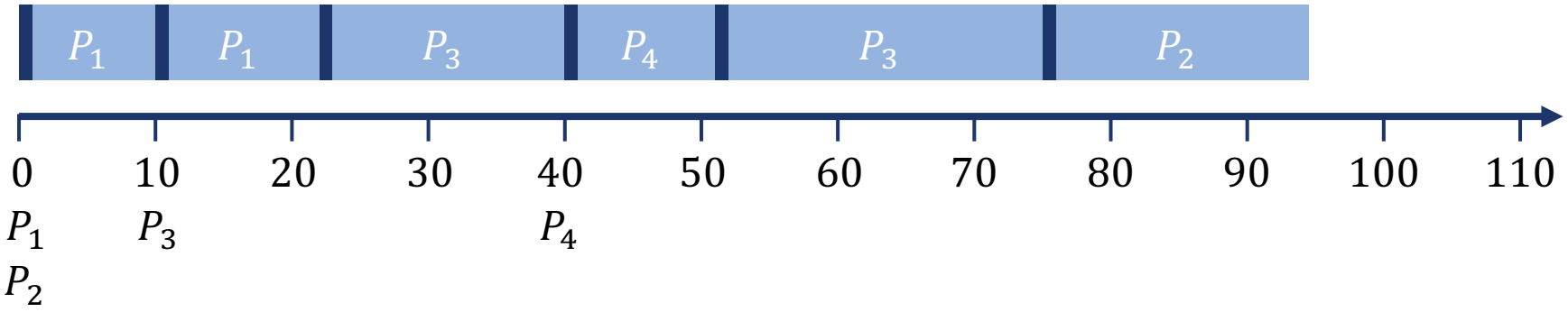


	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10
Deadline	35	120	80	60

	Start [ms]	Warte [ms]	End [ms]
P_1	1	2	22
P_2	76	76	96
P_3	23	24	75
P_4	41	1	51

	Durchsatz	AWT	ART	ATT
FCFS - 1	4/104	30,00 ms	30,00 ms	55,00 ms
FCFS - 2	4/104	32,50 ms	32,50 ms	57,50 ms
RR	4/110	42,25 ms	13,25 ms	67,25 ms
EDF	4/96			

Beispiel: EDF Scheduling

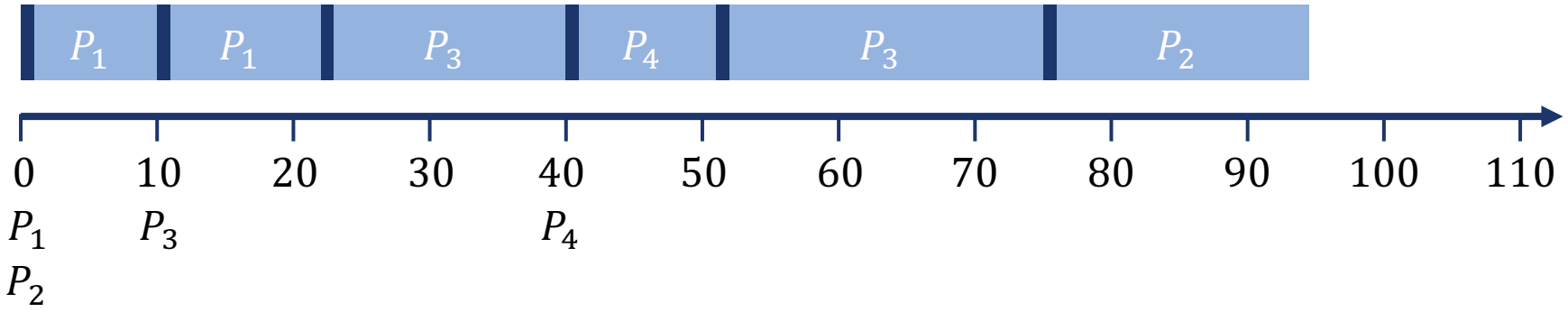


	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10
Deadline	35	120	80	60

	Start [ms]	Warte [ms]	End [ms]
P_1	1	2	22
P_2	76	76	96
P_3	23	24	75
P_4	41	1	51

	Durchsatz	AWT	ART	ATT
FCFS - 1	4/104	30,00 ms	30,00 ms	55,00 ms
FCFS - 2	4/104	32,50 ms	32,50 ms	57,50 ms
RR	4/110	42,25 ms	13,25 ms	67,25 ms
EDF	4/96	25,75 ms		

Beispiel: EDF Scheduling

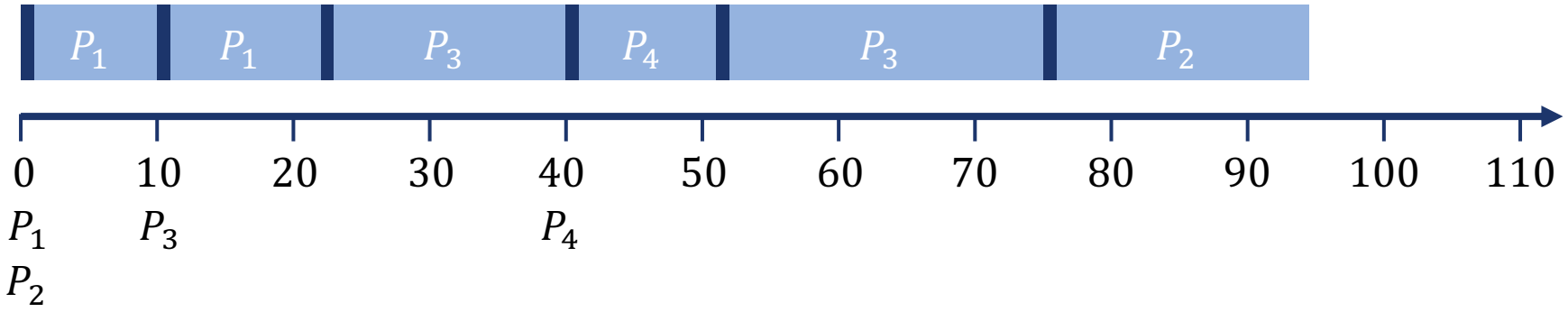


	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10
Deadline	35	120	80	60

	Start [ms]	Warte [ms]	End [ms]
P_1	1	2	22
P_2	76	76	96
P_3	23	24	75
P_4	41	1	51

	Durchsatz	AWT	ART	ATT
FCFS - 1	4/104	30,00 ms	30,00 ms	55,00 ms
FCFS - 2	4/104	32,50 ms	32,50 ms	57,50 ms
RR	4/110	42,25 ms	13,25 ms	67,25 ms
EDF	4/96	25,75 ms		

Beispiel: EDF Scheduling

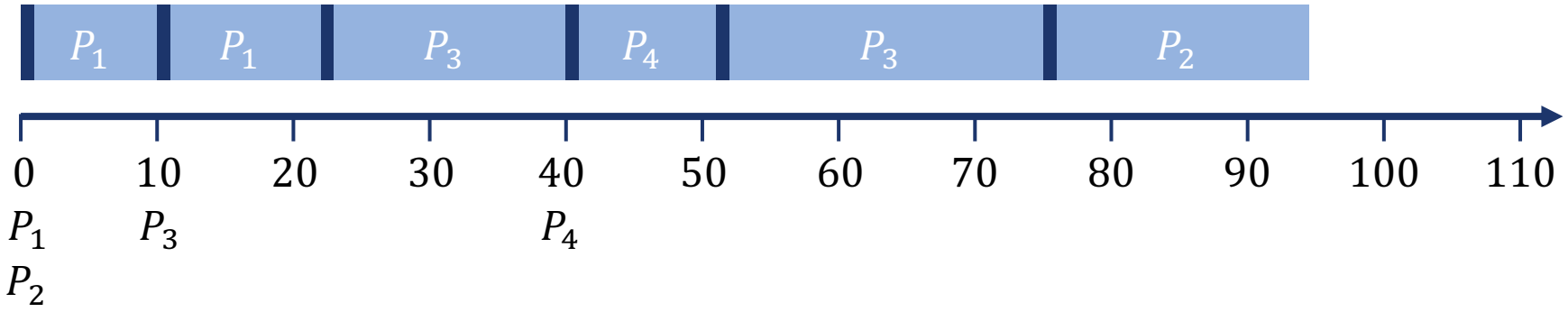


	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10
Deadline	35	120	80	60

	Start [ms]	Warte [ms]	End [ms]
P_1	1	2	22
P_2	76	76	96
P_3	23	24	75
P_4	41	1	51

	Durchsatz	AWT	ART	ATT
FCFS - 1	4/104	30,00 ms	30,00 ms	55,00 ms
FCFS - 2	4/104	32,50 ms	32,50 ms	57,50 ms
RR	4/110	42,25 ms	13,25 ms	67,25 ms
EDF	4/96	25,75 ms	22,75 ms	

Beispiel: EDF Scheduling

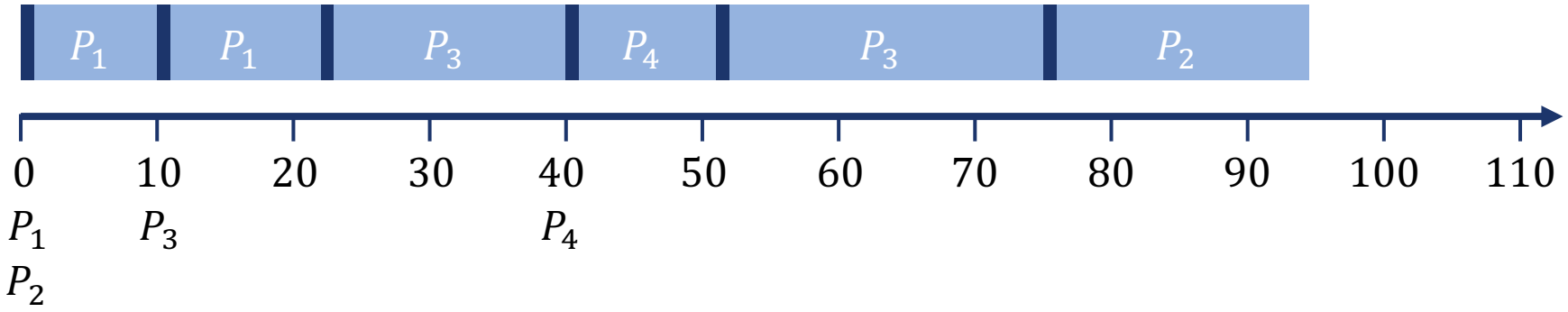


	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10
Deadline	35	120	80	60

	Start [ms]	Warte [ms]	End [ms]
P_1	1	2	22
P_2	76	76	96
P_3	23	24	75
P_4	41	1	51

	Durchsatz	AWT	ART	ATT
FCFS - 1	4/104	30,00 ms	30,00 ms	55,00 ms
FCFS - 2	4/104	32,50 ms	32,50 ms	57,50 ms
RR	4/110	42,25 ms	13,25 ms	67,25 ms
EDF	4/96	25,75 ms	22,75 ms	

Beispiel: EDF Scheduling



	P_1	P_2	P_3	P_4
Ankunftszeit	0	0	10	40
Arbeitszeit	20	30	40	10
Deadline	35	120	80	60

	Start [ms]	Warte [ms]	End [ms]
P_1	1	2	22
P_2	76	76	96
P_3	23	24	75
P_4	41	1	51

	Durchsatz	AWT	ART	ATT
FCFS - 1	4/104	30,00 ms	30,00 ms	55,00 ms
FCFS - 2	4/104	32,50 ms	32,50 ms	57,50 ms
RR	4/110	42,25 ms	13,25 ms	67,25 ms
EDF	4/96	25,75 ms	22,75 ms	48,50 ms

Earliest Deadline First

- Verhalten
  - Prozesse haben feste Endzeit
    - Sortierung nach Endzeit
    - Nächste Endzeit wird als erstes gewählt
  - Neuberechnung notwendig
    - Bei Aktivierung eines Prozesses
    - Terminierung des aktiven Prozesses
  - Sowohl präemptiv als auch nicht-präemptiv
- Vorteile
  - Potentiell optimales Scheduling
  - Einhalten von Deadline kann vorausgesagt werden
- Nachteile
  - Endzeit und Laufzeit muss bekannt sein
  - Prozesse müssen sortiert werden

Zusammenfassung

	FCFS	RR	EDF	FAIR
Nicht-Präemptiv				
Präemptiv				
Laufzeit				
Kooperatives Verhalten vorteilhaft				
Implementierungen				