オペレーティングシステム
第４章 スケジューリング

https://github.com/tctsigemura/OSTextBook

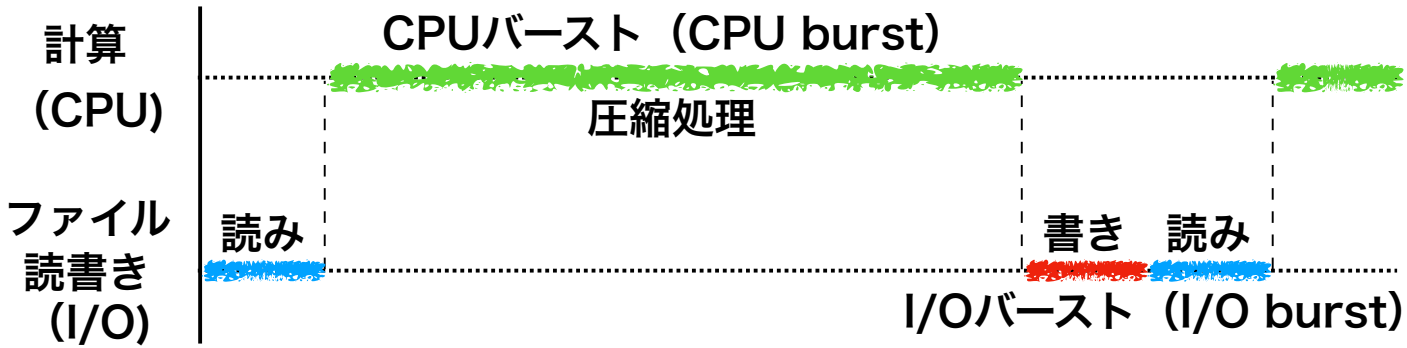
評価基準

- スループット（Throughput）
- ターンアラウンド時間（Turnaround time）
- レスポンス時間（Response time）
- 締め切り（Deadline）
- その他（公平性，省エネ，予測性など）

スケジューリングの目標

コンピュータの種類	重視する性能
メインフレーム（バッチ処理）	スループット，ターンアラウンド時間
ネットワークサーバ	レスポンス時間，スループット
デスクトップパソコン	レスポンス時間
モバイルデバイス	レスポンス時間，省エネルギー
組込み制御	締め切り

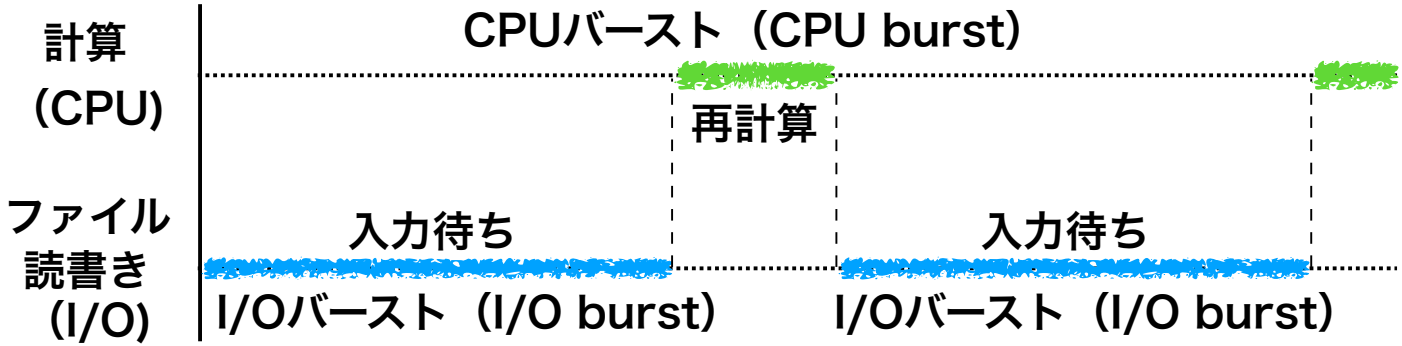
CPU バウンドプロセス



(a) CPUバウンド（CPU-bound）プロセス

- 動画圧縮の例
- I/O バウンドプロセス（エクセル）

I/O バウンドプロセス

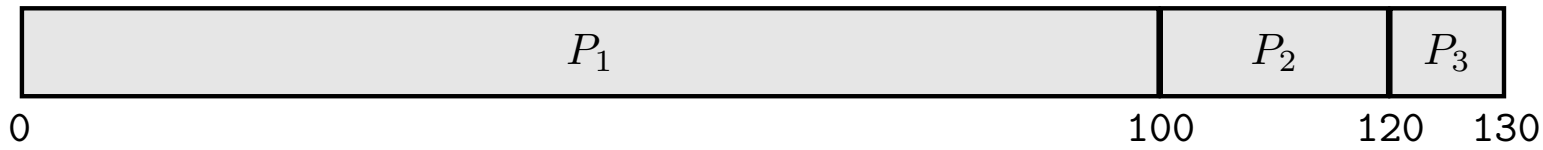


(b) I/Oバウンド（I/O-bound）プロセス

- スプレッドシートの例

FCFS スケジューリング（1）

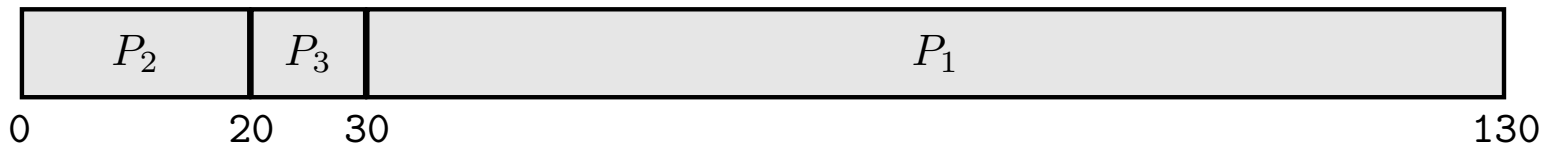
プロセス	到着時刻	CPU バースト時間 (ms)
P_1	0	100
P_2	0	20
P_3	0	10



- P_1，P_2，P_3 の順に実行
- 平均ターンアラウンド時間（$(100 + 120 + 130)/3 = 117$ ms）

FCFS スケジューリング（2）

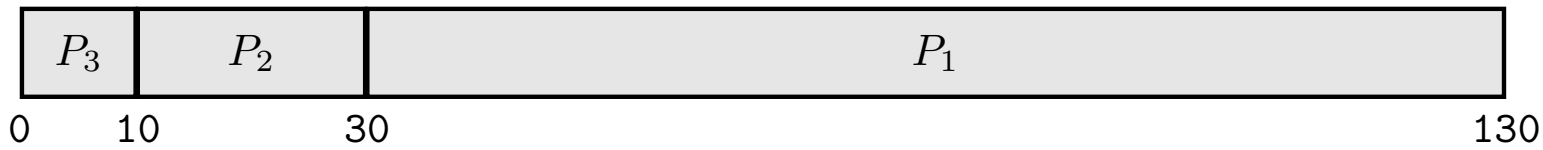
プロセス	到着時刻	CPU バースト時間 (ms)
P_1	0	100
P_2	0	20
P_3	0	10



- P_2，P_3，P_1 の順に実行
- 平均ターンアラウンド時間（$(20 + 30 + 130)/3 = 60$ ms）

SJF スケジューリング

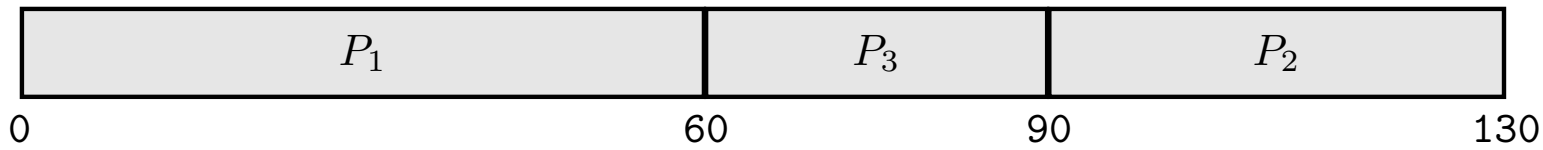
プロセス	到着時刻	CPU バースト時間 (ms)
P_1	0	100
P_2	0	20
P_3	0	10



- 平均ターンアラウンド時間（$(10 + 30 + 130)/3 = 57$ ms）

SJF スケジューリング（比較のため）

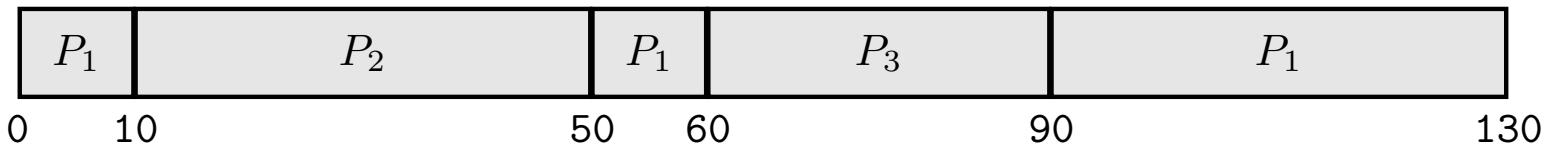
プロセス	到着時刻	CPU バースト時間 (ms)
P_1	0	60
P_2	10	40
P_3	60	30



- SJF はプリエンプションなし
- 平均ターンアラウンド時間
  $(((60-0)+(90-10)+(130-60))/3 = 70$ ms)

SRTF スケジューリング

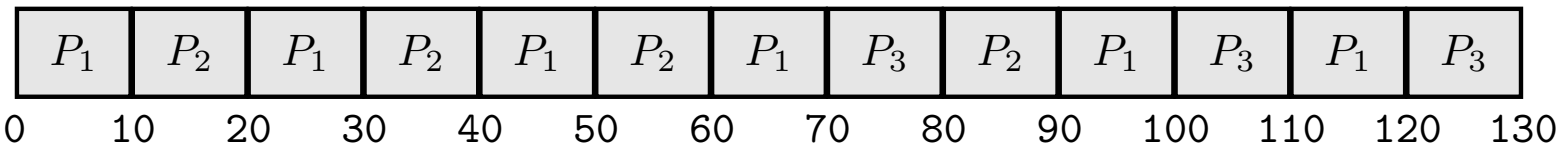
プロセス	到着時刻	CPU バースト時間 (ms)
P_1	0	60
P_2	10	40
P_3	60	30



- SRTF はプリエンプションあり
- 平均ターンアラウンド時間
  $(((130-0)+(50-10)+(90-60))/3 = 67$ ms$)$

RR スケジューリング（1）

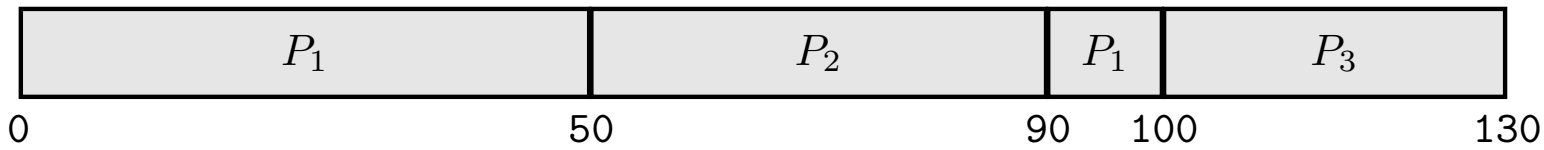
プロセス	到着時刻	CPU バースト時間 (ms)
P_1	0	60
P_2	10	40
P_3	60	30



- クォンタムタイム＝ 10ms
- 平均ターンアラウンド時間
  $(((120-0)+(90-10)+(130-60))/3 = 90)$

RR スケジューリング（2）

プロセス	到着時刻	CPU バースト時間 (ms)
P_1	0	60
P_2	10	40
P_3	60	30

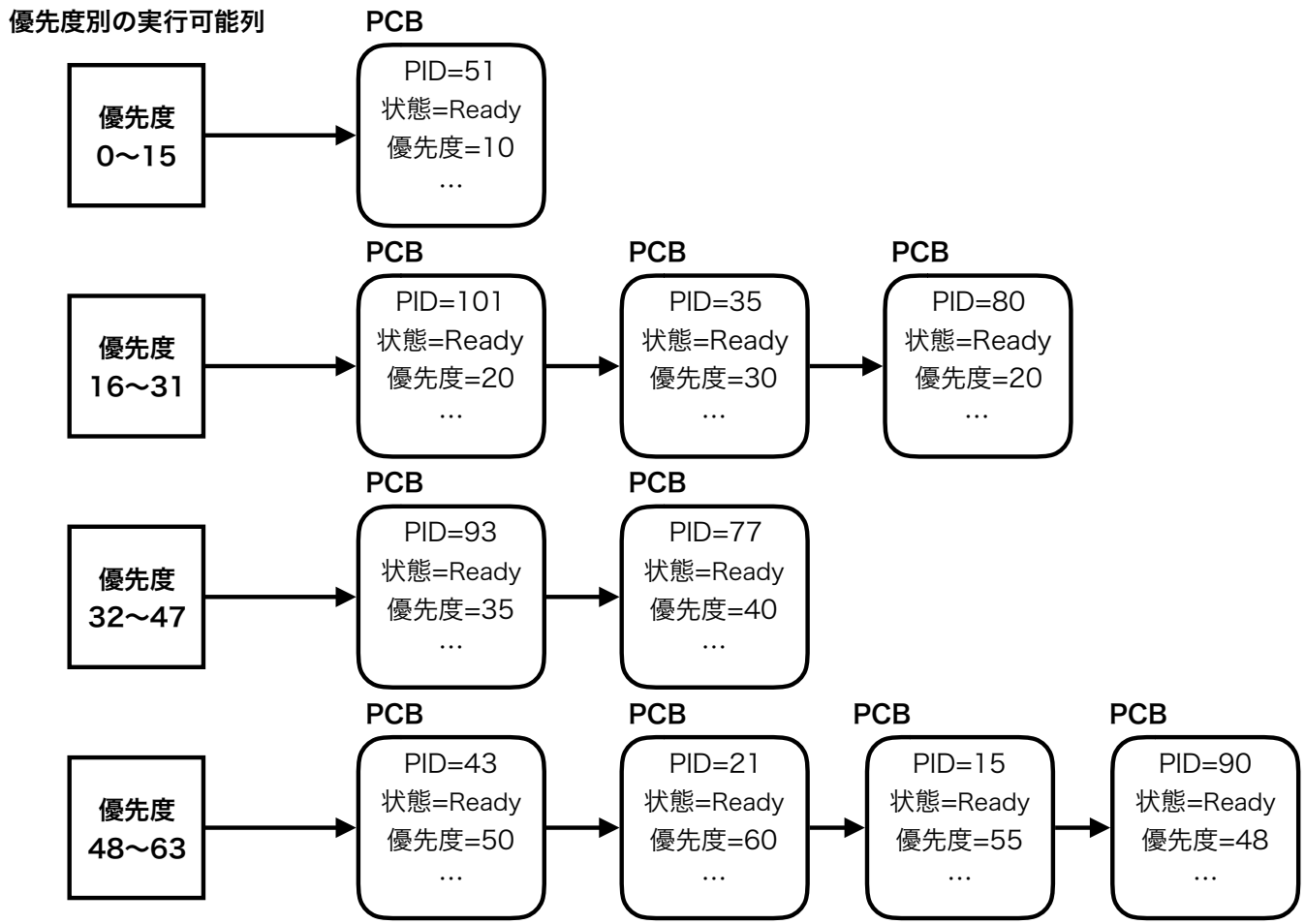


- クォンタムタイム＝ 50ms
- 平均ターンアラウンド時間
  $(((100-0)+(90-10)+(130-60))/3 = 83$ ms)

優先度順スケジューリング

- 静的・動的
- スタベーション
- エージング

FB スケジューリング



- エージング

TacOS のスケジューラ

```
// プロセスキューで p1 の前に p2 を挿入する   p2 -> p1
void insProc(PCB p1, PCB p2) {
  p2.next=p1;
  p2.prev=p1.prev;
  p1.prev=p2;
  p2.prev.next=p2;
}

// プロセススケジューラ:プロセスを優先度順で readyQueue に登録する
// (カーネル外部からも呼び出されるのでここで割込み禁止にする)
public void schProc(PCB proc) {
  int r = setPri(DI|KERN);                    // 割り込み禁止、カーネル
  int enice = proc.enice;
  PCB head = readyQueue.next;                 // 実行可能列から
  while (head.enice<=enice)                   //   優先度がより低い
    head = head.next;                         //     プロセスを探す
  insProc(head,proc);                         // 見つけたプロセスの
  setPri(r);                                  //   直前に挿入する
}                                             // 割り込み状態を復元する
```

TacOS の実行可能列（参考）

- `yield`
- `dispatch`
- 実行可能列