コンピュータ・サイエンス2

第4回
オペレーティングシステム

人間科学科コミュニケーション専攻
白銀 純子

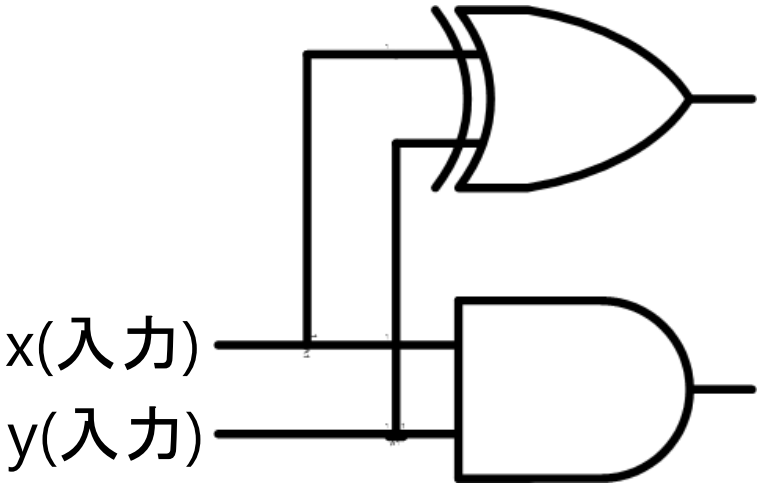
第4回の内容

- コンピュータの特長と実際
- オペレーティングシステム
- クラウドコンピューティング

前回の出席課題の解答

設問1

- 半加算器で入力xが1、入力yが0のとき、繰り上がり(c)と和(f)はどうなるかを答えなさい。



f(出力・xとyの和)

c(出力・xとyの和のうちの繰り上がり)

解答

- **繰り上がり(c): 0**
- **和(f): 1**

設問2

- 下記は、OS起動前とOS起動後のどちらに処理されるか答えなさい。
  a. どんなHDDやSSDが接続されているかのチェック
  b. プリンタのデバイスドライバを読み込み
  c. ファイルシステムを動作させるためのプログラムをメインメモリに登録
  d. CPUが接続されているかどうかの確認

解答

- **OS起動前: a, d**
- **OS起動後: b, c**

前回の質問の回答

半加算器と全加算器の違い

- 半加算器
  - 前の桁からの繰り上がりの加算をしない
  - 全加算器のベースとなっている
- 全加算器
  - 前の桁からの繰り上がりの加算をする

} 他の違いは特になし

前回の復習

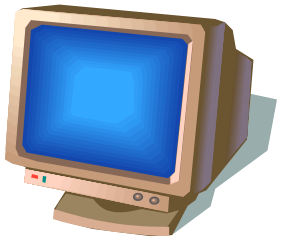
コンピュータの起動(p. 52)

- コンピュータの電源を入れて、デスクトップが画面に表示されるまでの処理

電源ボタンを押す

Windows

OSが起動する



デスクトップ画面が表示される

- BIOSが起動
- ビデオカード上のビデオROMが起動
- ブートディスクを識別

- カーネルが起動
- デバイスドライバを読み込み
- ログオン画面を表示

※OS(詳しくは後日)

- ハードウェアを効率的に動かすためのソフトウェア
- アプリケーション(人間が操作するソフトウェア)とハードウェアとの仲介

OS起動までの処理[1](p. 52)

1. マザーボードのBIOS-ROMにより制御開始
   - BIOS: 各種装置に記録されているソフトウェアで、それぞれの装置の初期設定や制御などを担当
   - BIOS-ROM: BIOSを記録している部品

単純に言うと、マザーボードが動作開始

2. マザーボードのBIOSが起動し、各種装置を制御
   - CPUのテスト(故障などの不具合がないか)
   - メインメモリのテスト(故障などの不具合がないか)
   - チップセットや各種ポートの設定
     - チップセット: マザーボード上で、CPUやメインメモリ、拡張カードなどの間でデータをやり取りを管理するための回路

単純に言うと、マザーボードに取り付けられているデバイスのチェック

OS起動までの処理[2](p. 52)

3. ビデオカード上のビデオROMが起動
   - ビデオROM: グラフィックスボードのBIOS

単純に言うと、ビデオカードが動作開始

4. マザーボードのBIOSによる処理内容をビデオROMが受け取って表示
   - メインメモリに対するテストの結果
   - サウンドボードやLANボードの設定状況
   - ドライブやディスクの設定状況
   - etc.

マザーボードのBIOSの処理結果(ビデオROMの処理ではない)

単純に言うと、マザーボードでの処理結果をディスプレイに表示開始

OS起動までの処理[3](p. 52)

5. ブートするディスクを識別
   - ブート: コンピュータを起動すること
   - ブートするディスク: OSを記録しているHDDやSSD
     - コンピュータにHDDやSSDを複数取り付けることが可能
     - OSを記録していないHDDやSSDを使ってブートすることは不可能

単純に言うと、OSを記録しているHDDやSSDを検索

6. OSの起動プログラムを呼び出して制御を受け渡し

単純に言うと、マザーボードでの処理を終了し、OSを呼び出し

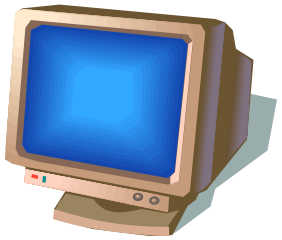
コンピュータの起動(p. 52)

- コンピュータの電源を入れて、デスクトップが画面に表示されるまでの処理

電源ボタンを押す

Windows

OSが起動する



デスクトップ画面が表示される

- BIOSが起動
- ビデオカード上のビデオROMが起動
- ブートディスクを識別

- カーネルが起動
- デバイスドライバを読み込み
- ログオン画面を表示

※OS(詳しくは後日)

- ハードウェアを効率的に動かすためのソフトウェア
- アプリケーション(人間が操作するソフトウェア)とハードウェアとの仲介

デスクトップの表示まで[1](p. 52)

1. カーネルの起動
   - カーネル: OSの中でも最も中核となる重要な部分
     - プロセスの実行制御やメモリ管理
     - ファイルシステムの管理
     - 主記憶装置へのプログラムの登録
     - etc.

単純に言うと、OSが動作開始

デスクトップの表示まで[2](p. 52)

2. デバイスドライバの読み込み
   - デバイスドライバ: 補助記憶装置や入出力装置などの制御を行うソフトウェア
     - HDDやグラフィックスボード、サウンドボード、LANボードなどそれぞれの装置にデバイスドライバは必要

単純に言うと、取り付けられているデバイスを利用可能に準備

3. ログオン画面を表示
   - 利用者がユーザ名とパスワードを入力することでデスクトップが表示

プログラムの起動と実行

ソフトウェアの起動[1](p. 53)

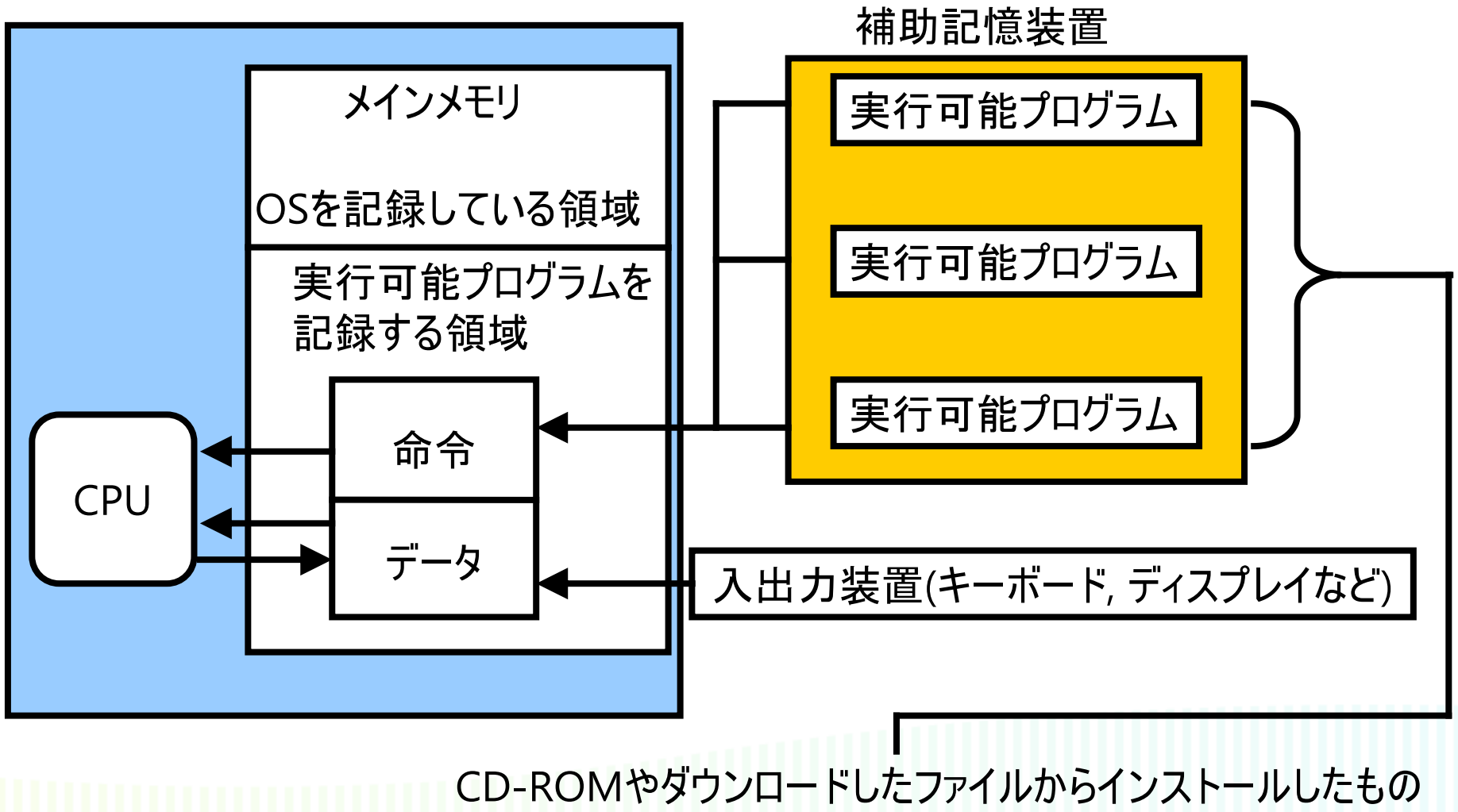
- ソフトウェアの起動アイコンをダブルクリックor 実行可能プログラムを指定
  - 起動アイコンは、ソフトウェアの実行プログラムそのもの、または実行プログラムのありかを示したもの
    →ダブルクリックすることは、実行可能プログラムの指定と同じ
- 実行可能プログラムがHDDやSSDからメインメモリに転送
  - 「ローディング」と呼ぶ

実行可能プログラム: コンピュータが直接命令を読んで実行できるプログラム

- 人間が作成したプログラムは、そのままではコンピュータが直接読んだりできないので、実行可能プログラムに翻訳する(詳細は後日)

ソフトウェアの起動[2](p. 53)

- ローディングされたプログラムの命令をCPUが取り出して解読

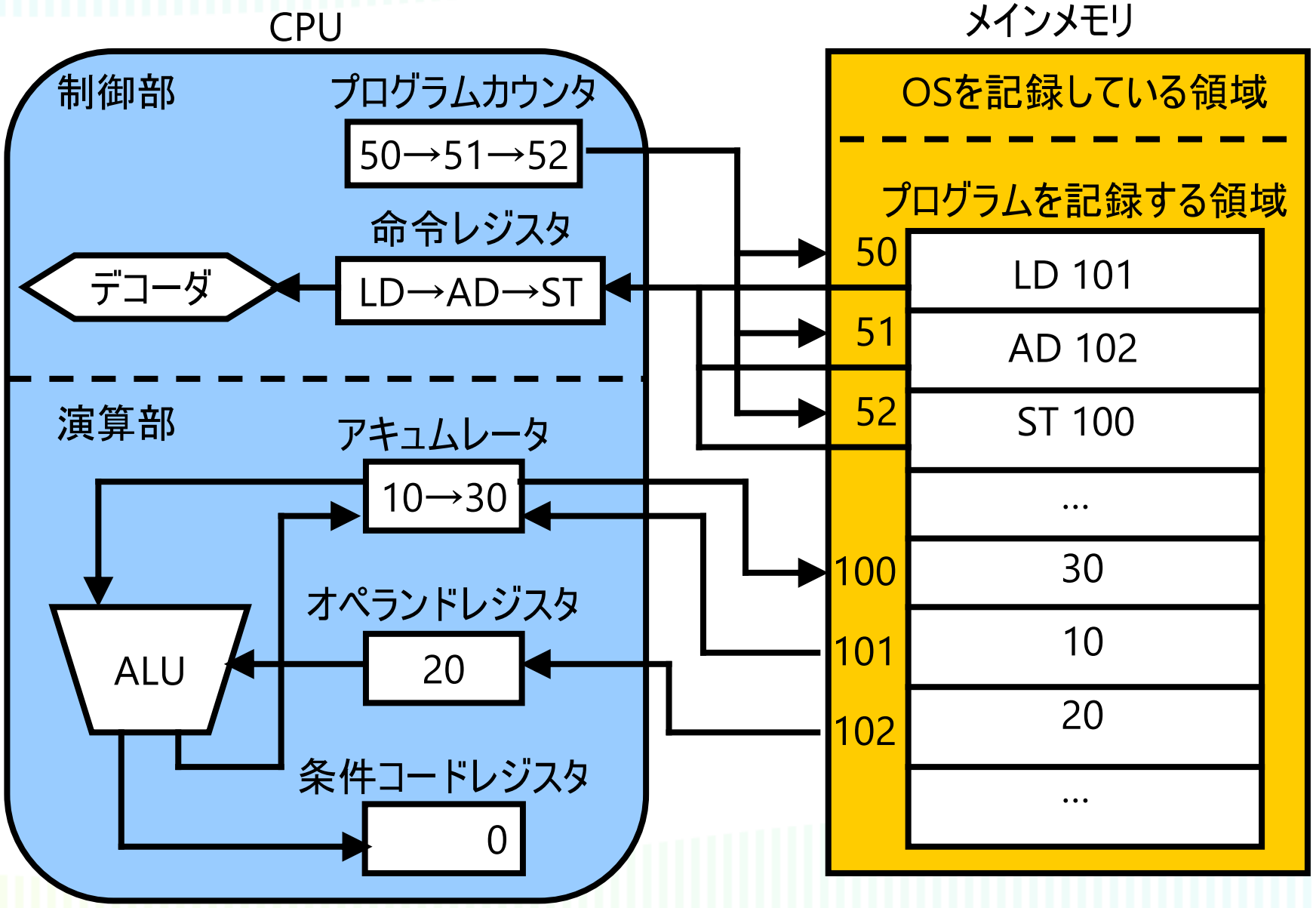


CPUの命令解読例(p. 54)

- 「10+20」の計算をするプログラム(C言語)
  - x, y, zという名前の3つの箱を用意する
    - プログラムでは、データは箱に入れて扱う
  - yとzという箱にそれぞれ10と20を入れる
  - xという箱に、「y+z」の結果を入れる
  - 計算結果を画面に表示する

```
#include<stdio.h>
void main() {
    int x;
    int y = 10;
    int z = 20;
    x = y + z;
    printf("x = %d¥n", x);
}
```

CPUの命令解読例[手順図](p. 54)



CPUの命令解読例[手順1](p. 54)

1. プログラムカウンタによって、メインメモリの50番地が設定される
   - これにより、50番地の「LD 101」命令が取り出され、命令レジスタに転送される
2. プログラムカウンタの数値を1増やす
3. デコーダが「LD 101」を解釈して実行する
   - これにより、101番地のデータをアキュムレータに転送する(「LD」はCPUへのデータの転送命令)

CPUの命令解読例[手順2](p. 54)

4. プログラムカウンタの数値が51なので51番地の命令を取り出す
   - これにより、51番地の「AD 102」命令が取り出され、命令レジスタに転送される
5. プログラムカウンタの数値を1増やす
6. デコーダが「AD 102」を解釈して実行する
   - これにより、102番地のデータをオペランドレジスタに転送する
   - ALUにより加算処理が行われる(「AD」は加算命令)
     - 「10 + 20 = 30」が実行される
   - 計算結果「30」がアキュムレータに一時的に格納される

CPUの命令解読例[手順3](p. 54)

7. プログラムカウンタの数値が52なので52番地の命令を取り出す
   - これにより、52番地の「ST 100」命令が取り出され、命令レジスタに転送される
8. プログラムカウンタの数値を1増やす
9. デコーダが「ST 100」を解釈して実行する
   - これにより、アキュムレータにあるデータ「30」を100番地に格納する(「ST」はデータのメインメモリへの格納命令)
10. 次の命令へ進む

CPUの命令解読例[概要](p. 54)

- C言語のプログラムの「x=y+z;」の計算をするためにStep1～Step9の手順
  - Step1～Step4: 「y」という箱からデータを取り出す処理
  - Step5～Step6: 「z」という箱からデータを取り出し、「y+z」の計算をする処理
  - Step7～Step9: 「x」という箱に「y+z」の結果を入れる処理

Question!

コンピュータの特徴

コンピュータの特徴(p. 56)

- コンピュータでのプログラムの処理方式
  - デジタル方式
  - プログラム内蔵方式
  - 逐次制御方式

デジタル方式(p. 56)

- 全てのものを2進数に変換して扱う方式
  - プログラム
  - 入力・出力データ
    - 数字, 文字, 記号
    - 図表, 静止画, 動画, 音声, 音楽
    - etc.
- 論理素子・記憶素子: CPUやメモリの構成要素
  - 電圧の高低で動作を制御
    - 電圧の高低を2進数の0と1に対応

プログラム内蔵方式[1](p. 56)

- 当初のコンピュータ: 現在プログラムが行っている処理をスイッチと配線の組み換えでその都度操作
  - スイッチと配線の組み換えで様々な命令を表現
  - 非常に手間がかかり、間違いが多発
    - たくさんのスイッチのONとOFFを逐一切り替える必要
    - 同じ処理をするために毎回プログラムを作成する必要

プログラム内蔵方式[2](p. 56)

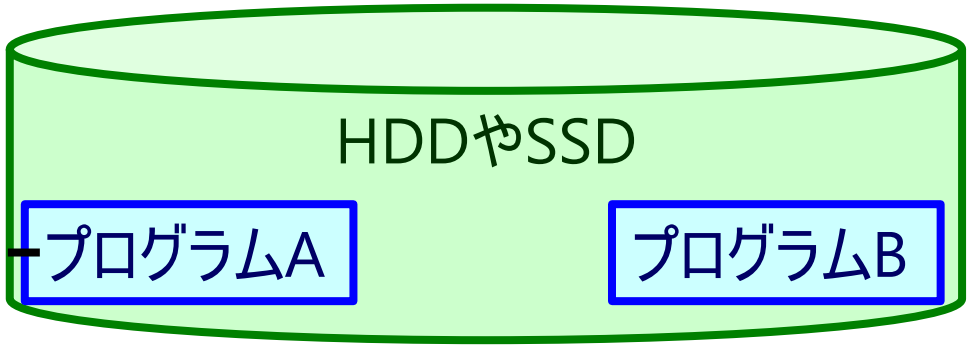
- プログラム内蔵方式: 作成したプログラムを記憶装置に記憶させて利用する方式
  - プログラムを実行するときに命令を記憶装置から取り出して実行
    - 同じ処理をするためにプログラムは1度だけ作成
  - 用途に応じてプログラムを入れ替え、様々な処理を実行

「ノイマン」という人によって提案された方式

- ノイマン:「コンピュータの父」と呼ばれるアメリカの数学者
- プログラム内蔵方式で動くコンピュータを「ノイマン型コンピュータ」
  - ✓ 現在のコンピュータのほとんどはノイマン型コンピュータ

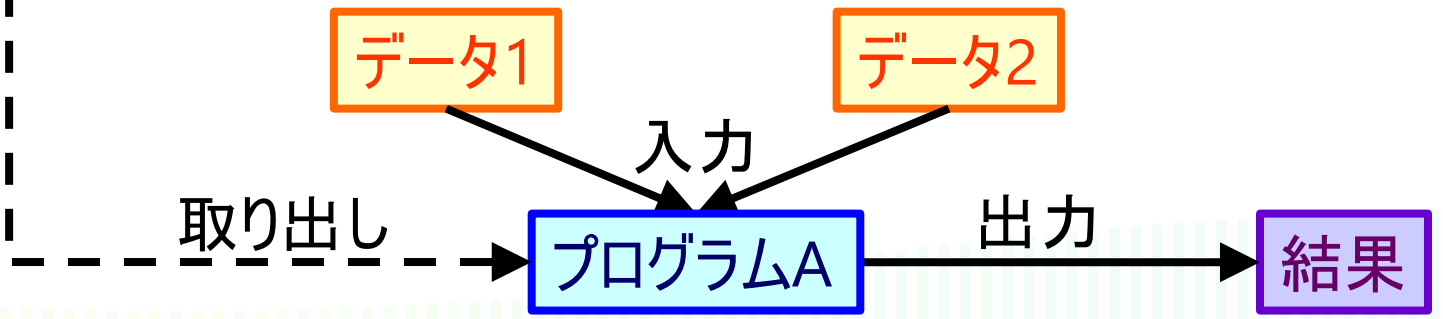
プログラム内蔵方式[3](p. 56)

普段: プログラムをHDDやSSDに記録



プログラム実行時: HDDやSSDから取り出して利用

- 処理に利用するデータを入力



逐次制御方式(p. 56)

- メインメモリから命令が1つずつ取り出されて実行される方式
  = メインメモリとCPUは頻繁にやりとり
  - プログラムカウンタによって、命令が格納されている番地を指定
  - その番地から命令を取り出して実行
  - プログラムカウンタの数値を1増やして次の命令の番地を指定

CPUとメインメモリの間の通信経路の転送速度のためにコンピュータ全体の処理の速度が遅くなる問題も存在

フォン・ノイマン・ボトルネック

Question!

コンピュータの実際

コンピュータの分類

- 筐体の大きさから分類すると…
  - スーパーコンピュータ
  - 汎用コンピュータ
  - サーバコンピュータ
  - パーソナルコンピュータ
  - モバイルコンピュータ

スーパーコンピュータ

- 一般的な汎用コンピュータよりも、演算速度が大幅に高速
- 利用されている分野
  - 気象学
  - 天文学
  - 流体力学
  - 金融工学, etc.

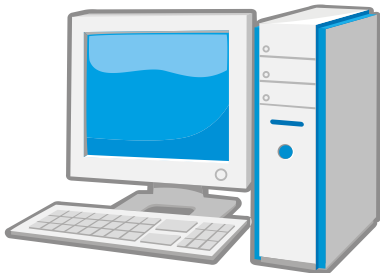
  大規模な数値解析によるシミュレーション用の科学技術計算
- 最近: グリッドコンピューティング
  - インターネットなどの広域ネットワークを利用し、各地のスーパーコンピュータを連結させて動作させる仕組み

サーバコンピュータ

- クライアントサーバシステムを実現するためのサービスを提供するコンピュータ
  - クライアント: サーバに処理を要求するコンピュータ
  - サーバ: クライアントから要求された処理を行い、結果をクライアントに返すコンピュータ
  - インターネットで広く利用
- サービスを提供するソフトウェアの種類により分類
  - Webサーバ
  - メールサーバ, etc.

パーソナルコンピュータ(PC)

- 個人の利用を前提としたコンピュータ
  - 大きさ、価格、性能などが個人向け
  - 以前は個人の趣味で利用、現在はビジネスでも利用
  - クライアントサーバシステムでのクライアントの役割
- 筐体の大きさや形状で分類
  - デスクトップPC
  - ノートPC



デスクトップPC
(タワー型)

デスクトップPC
(一体型)

ノートPC

モバイルコンピュータ

- 移動しながら使用できるコンピュータ
  - 携帯電話(スマートフォンを含む)
  - タブレット端末, etc.

オペレーティングシステム

ソフトウェアの種類(p. 64)

- オペレーティングシステム(OS)
  - ハードウェアを動かすための基本的なソフトウエア
    - ハードウェアと応用ソフトウェアの間のやり取りを仲介
    - ハードウェアの制御
- アプリケーション(アプリケーションソフト、とも)
  - OSをもとに動くソフトウェア
  - 人間が直接操作するソフトウェア
    - 文書処理ソフト
    - 表計算ソフト
    - Webブラウザ
    - etc.

オペレーティングシステム(p. 64)

- Operating System(OS)
- 「基本ソフトウェア」とも呼ばれる
- コンピュータに必須の基本的なソフトウェア
  - 役割: ハードウェア資源(CPUや記憶装置など)とソフトウェア資源(プログラムやデータ)を効率的に管理

※プログラム: コンピュータに指示をするための命令の集まり

OSの発展[1](p. 65)

- 道具や機械で計算をさせるという考えは古くから存在
  - 紀元前: 算盤
  - 1640年代: 歯車式計算機
    - フランスのパスカル(Blaise Pascal)が考案
  - 1945年: ENIAC(Electronic Numerical Integrator and Computer)
    - ペンシルベニア大学で開発

- 「OS」と呼べるほどのきちんとした形のソフトウェアはなかった
- 人間が線をつなぐことにより、計算していた
  - ✓ 計算速度は人間よりも高速
  - ✓ 異なる手順で計算をする場合には、配線をし直す必要

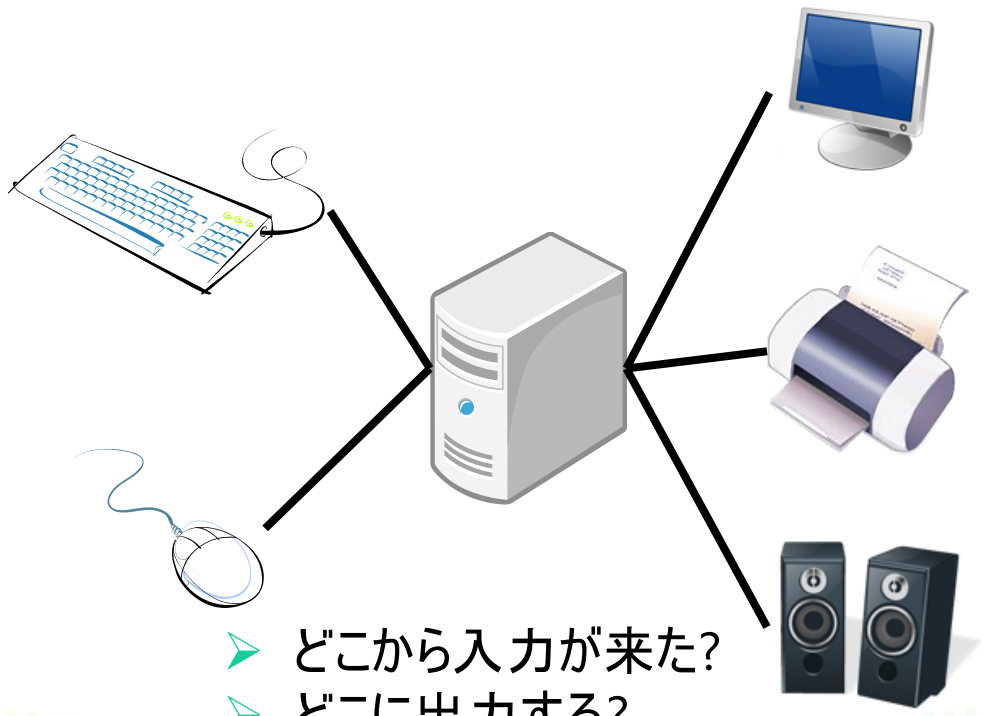
OSの発展[2](p. 65)

- ノイマン(John von Neuman)がプログラム内蔵方式を提唱
  - ハンガリー出身のアメリカの数学者
- 1949年: EDSAC(Electronic Delay Strage Automatic Calculator)
  - Maulice Wilkesらが開発
  - 実際に動くことはなかったが、プログラム可能なコンピュータの原型

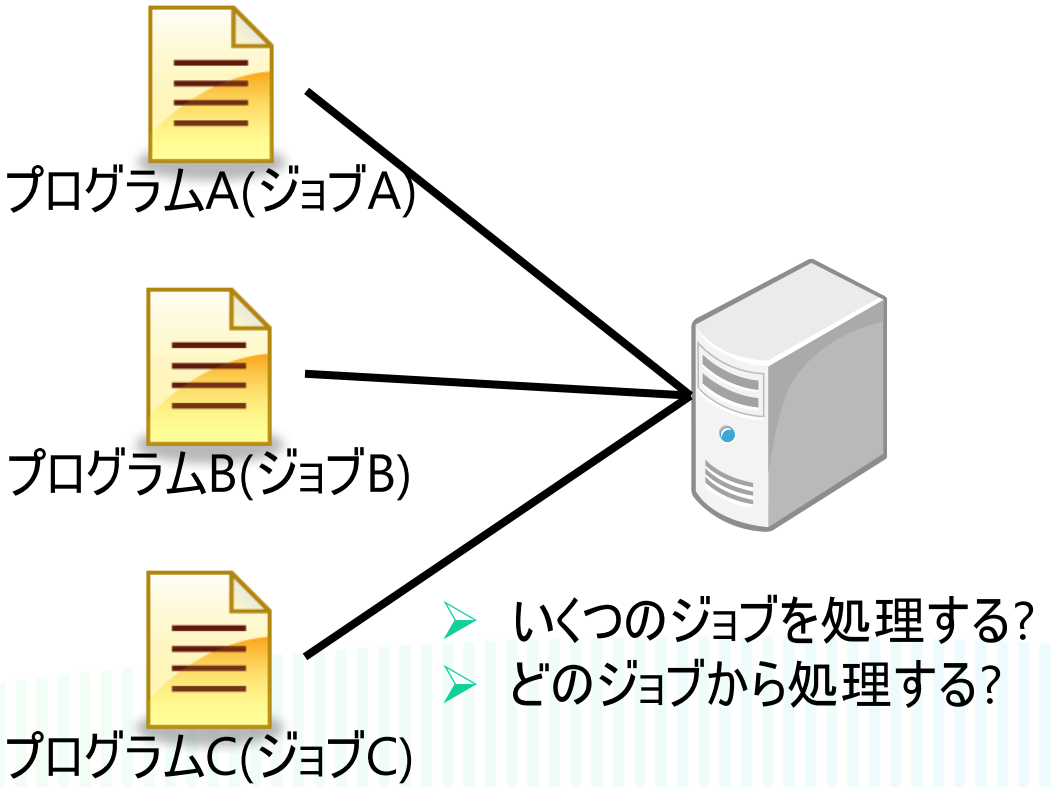
- 初期(この頃)のコンピュータでは、実行できるプログラムは1つだけ

OSの発展~OSでの処理~[1](p. 66)

- 時代が進むと…
  - 1台のコンピュータに様々な入力・出力装置(入出力制御)
  - 複数のプログラム(ジョブ)を連続して処理(ジョブ管理)
    - ジョブ: 人間がコンピュータに命令する処理の単位



- どこから入力が来た?
- どこに出力する?



- いくつのジョブを処理する?
- どのジョブから処理する?

OSの発展～OSでの処理～[2](p. 67)

- プログラム(コンピュータへの命令書)を記述できる言語
  - 初期: 機械語(2進数だけの表現)で記述
  - 機械語は不便: アセンブリ言語が登場
    - アセンブリ言語でプログラムを書き、機械語に翻訳(アセンブラ)

  - その後: FORTRAN(FORmula TRANSlator)が登場
    - 英語に近い形で記述し、機械語に翻訳(コンパイラ)

OSの発展～OSでの処理～[3](p. 67)

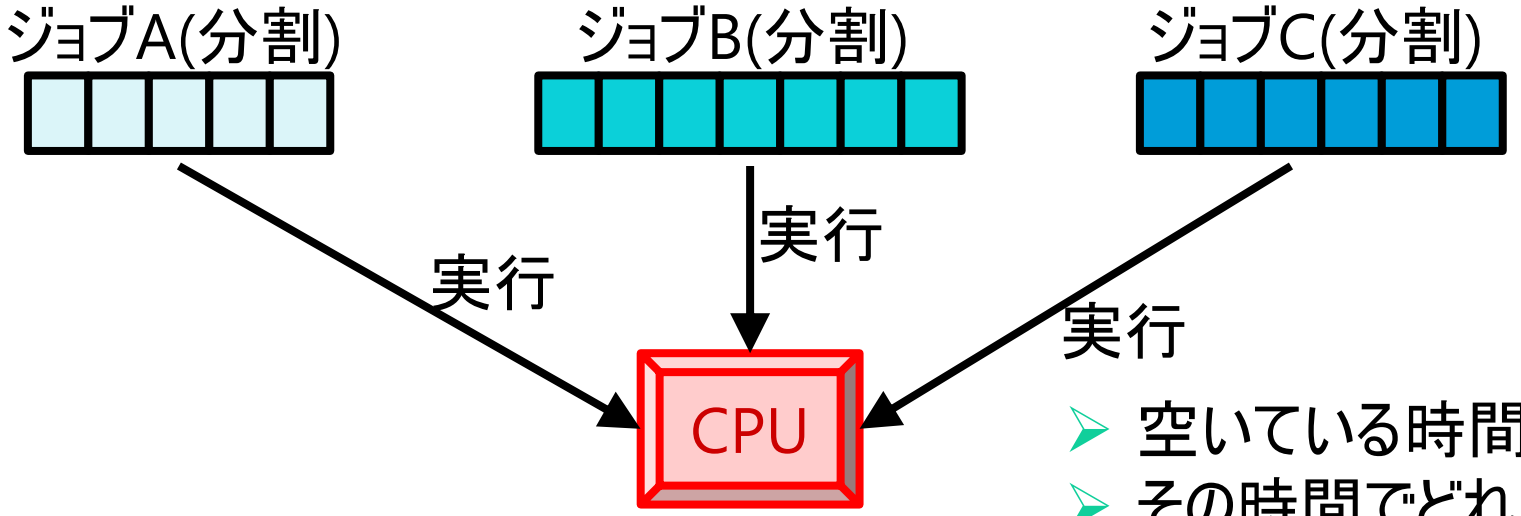
- 当時のコンピュータ: メインメモリの容量が少
  - メインメモリ: プログラム実行時に、プログラム内の命令や利用するデータを一時的に保存
    - 容量が少ないと、命令の多いプログラムやサイズの大きいデータを保存しきれない

➡ メインメモリが足りなくなったら、補助記憶装置(HDD)にメインメモリの役割をさせる(仮想記憶装置(Virtual Storage))

➡ メインメモリと補助記憶装置(仮想記憶装置)とのやりとりを管理するプログラムも必要

OSの発展～OSでの処理～[4](p. 68)

- 様々な処理をコンピュータで行う必要
  - CPUに、効率よく、各ジョブを実行させる必要
    (タイムシェアリングシステム (Time Sharing System, TSS))



- 空いている時間はいつ?
- その時間でどれを実行する?

➡ 多重プログラミング(同時に複数の処理を並行して行うこと)が可能に
- CPUが一度に行う処理は1つだが、CPUの処理時間を細かく分割し、複数の処理を同時に行っているように見せかける方法

➡ CPUの割り当て制御プログラムが必要

OSの発展～OSでの処理～[5](p. 68)

- 1台のコンピュータを遠隔地から使う必要

  ➡ リモートアクセスのプログラムが必要

OSの発展～OSでの処理～[6](p. 68)

- DOS/VS(Disk Operating System/Virtual Strage)にこれまでの必要なプログラムがすべて搭載
  - コンピュータがメインフレームと呼ばれていた頃に搭載されていたOS

- コンピュータを制御するための各種プログラムが開発
- さまざまなプログラムや処理方式を統合し、1つにまとめて提供するという方向性

OS(Operating System)として登場

OSの発展～その後～[1](p. 69)

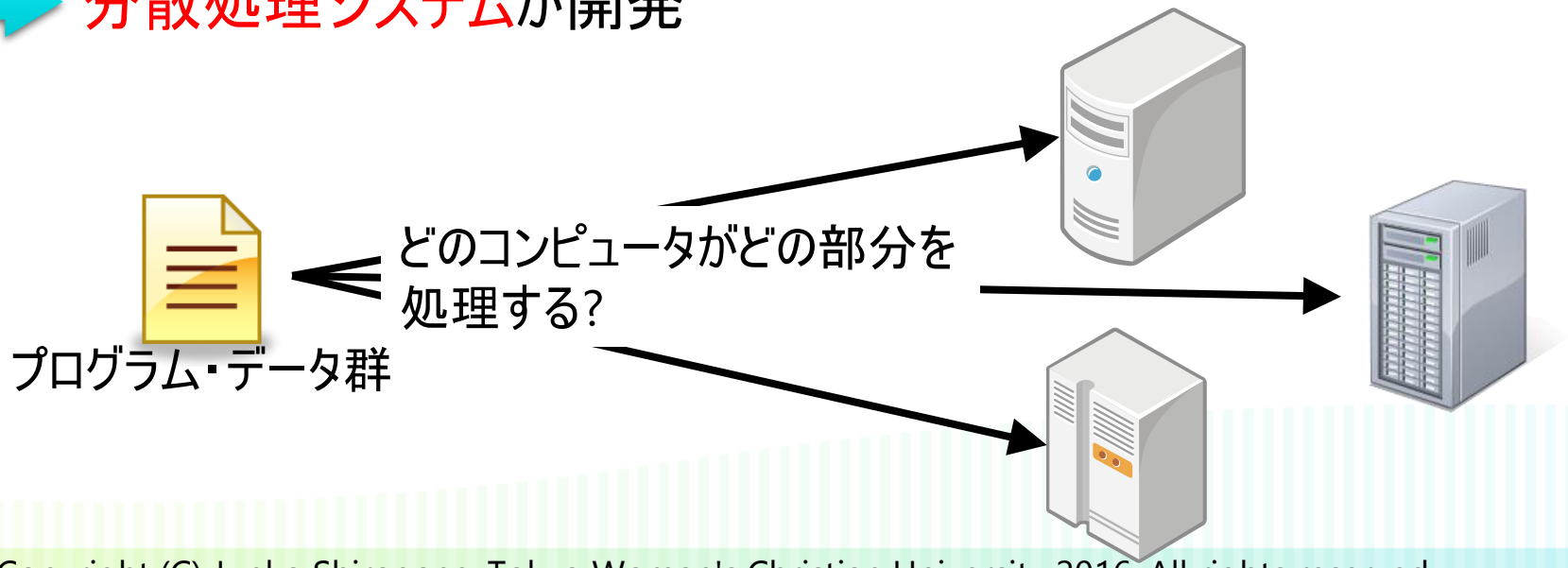
- 1台のコンピュータを数台分のコンピュータとして利用する考え方が登場(仮想化)
  - タイムシェアリングシステムの発展
  - CPUやメインメモリなどを仮想化

  ➡ マルチプロセッシングシステムや複数のOSを管理するシステムが必要
  - ➢ マルチプロセッシングシステム: 複数個のCPUなどを制御するためのシステム

- コンピュータとコンピュータをつないで利用する考え方が登場(コンピュータネットワーク)

OSの発展～その後～[2](p. 69)

- 離れたところにあるコンピュータ同士を専用回線でつなげて使う必要
  - ➡ 通信制御装置を管理するプログラムが必要
- コンピュータとコンピュータをつないで利用する考え方が登場
  (コンピュータネットワーク)
  - 1台に処理がかたより、処理速度が遅くなってしまうため
    - ➡ 分散処理システムが開発



現在(p. 70)

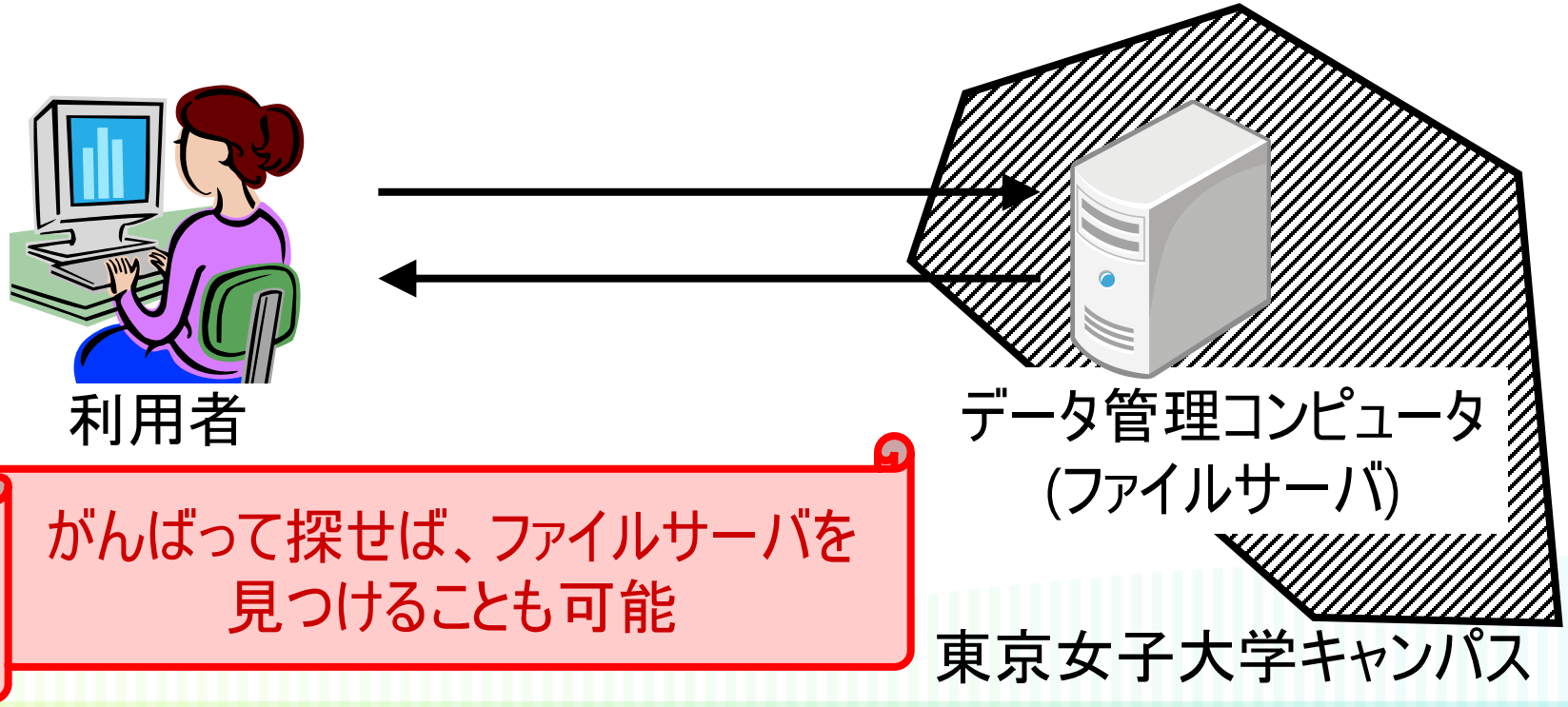
- メインフレームの時代のOSの機能は、現在でも利用
- クラウドコンピューティングの登場(「クラウド」とも)
  - ソフトウェアのオープンソース化や無償化
    - オープンソース: 機械語に翻訳前のプログラムを公開して誰でも編集可能にしたもの
  - 企業や組織などで利用するソフトウェア・ハードウェアの維持管理コストの軽減のニーズ

クラウドコンピューティング[1](p. 70)

- これまで: 利用者がアクセスするコンピュータは1台
  - データ管理, メールの処理, etc.
  - どこにあるどのコンピュータか、特定可能

Ex. 東京女子大学でのホームのデータ利用

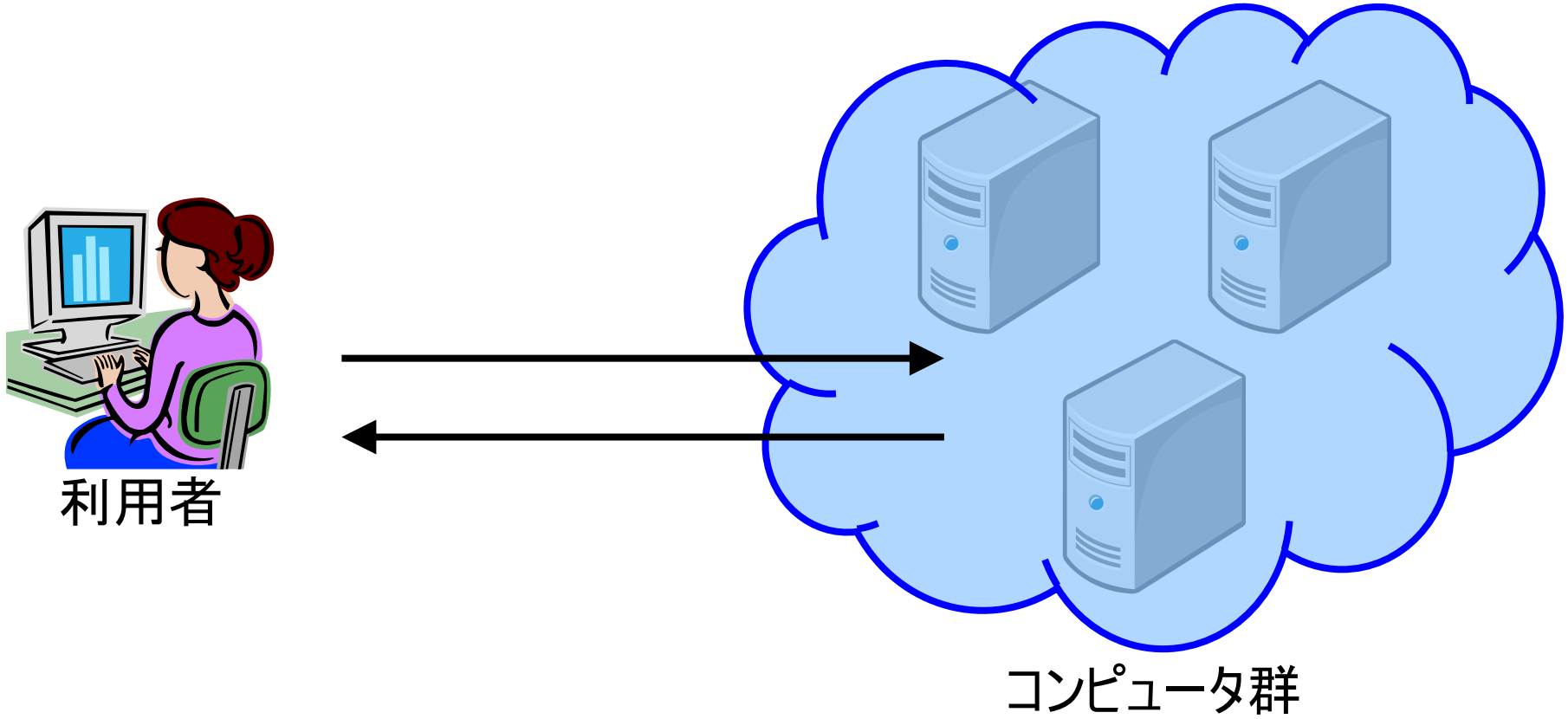
➢ データはファイルサーバに保存されているので、ネットワークを通じてどのコンピュータからも利用可能



がんばって探せば、ファイルサーバを
見つけることも可能

クラウドコンピューティング[2](p. 70)

- クラウド: たくさんのコンピュータのどれかとやりとり
  - たくさんのコンピュータを1つのコンピュータとして見せかけ



クラウドコンピューティング[3](p. 70)

- クラウドコンピューティングでは…
  - アクセスするコンピュータは、地理的に一か所に固まって置かれておらず、世界中の各地に置かれていることも

利用者側からすると…
自分が通信しているサーバが、日本にあるかアメリカにあるかもわからない

- アクセスするコンピュータの実体がどこにあるかわからない
- インターネットの概念図は、ネットワークを雲の形で表現することが多い

「クラウド(雲)コンピューティング」と呼ぶ

クラウドコンピューティングの形態(p. 71)

- SaaS(Software as a Service)
  - 必要なときにインターネットを通じてソフトウェアを利用できる仕組み
    - これまで: ソフトウェアは1台1台の利用者用のコンピュータにインストールして利用
  - 有料のものも無料のものも存在
- PaaS(Platform as a Service)
  - OSやデータベースなどの基本的なソフトウェアをインターネットを通じて利用できる仕組み
- IaaS(Infrastructure as a Service)
  - コンピュータの基盤を提供するサービス
    - 機材や回線、OSなどのシステムに必要なインフラ

Question!

仮想化

仮想化(p. 72)

- 仮想化: ハードウェアのリソースを複数台分に、複数台を1台分に見せかけて動作させること
  - リソース: CPUやメインメモリ、補助記憶装置、入出力装置などなどの利用可能な資源
  - 利用者それぞれの専用のリソースがあるように見せかけて利用者に使わせることが可能

仮想化の方法[1](p. 72)

- CPU
  - 複数の実行させるプログラム(プロセス)の制御
    - 起動の制御, 実行順序の管理, etc.
  - スレッドによる並行処理
    - スレッド: プロセスでの様々な処理を同時に実行させる方法
- メインメモリ
  - 仮想記憶装置の作成
  - メインメモリと仮想記憶装置の統合的な管理

仮想化の方法[2](p. 72)

- HDD/SSD
  - ファイルやフォルダ(ディレクトリ)の管理(ファイルシステム)
    - ファイルの作成・削除・ファイル名によるアクセス・ファイルの保護, etc.
- 入出力機器
  - 入力機器からの入力、結果の出力機器への出力の制御
    - プログラム制御方式: 入出力のデータ転送をCPUが管理
    - チャネル制御方式: 入出力のデータ転送を「チャネル」という専用装置が管理 (CPUを介して転送)
    - DMAコントローラ方式:入出力のデータ転送を「DMAコントローラ」という専用装置が管理 (CPUを介さないで転送)

OSの種類

以前のコンピュータ[1](p. 73)

- メインフレームの時代: 機種に依存(機種ごとに異なる)
  - 時代が進んでコンピュータ同士を接続して利用
    - ➡ OSが違うと、接続プログラムが非常に複雑化
- 現在: ある程度OSが標準化
  - 各種OSでサーバ用OSとクライアント用OSに分けられている
    - 同じハードウェアのコンピュータでも、どちらをインストールするかでサーバになったりクライアントになったり
    - 店で売られているPCはクライアント用OS

PCのOS[2](p. 73)

- MS-DOS: PCの初期の頃によく使われていたOS
  - Microsoft-Disk Operating System
  - 「コマンドプロンプト」が画面上に表示
    - コマンド(命令)を入力することで、OSに直接命令可能(マウスは使わず、キーボードから命令を入力するだけ)
  - CUI(Character User Interface)のOS
    - CUI: 人間とのやりとりの接点(User Interface)が文字だけ
- Mac OS: Apple社が開発したOS
  - GUI(Graphical User Interface)を初めて搭載したOS
    - GUI: マウスを使ってアイコンや、ボタン、入力フィールドなど、視覚的な操作が可能なユーザインタフェース(現在の形)

現在のOS[1](p. 74)

- Mac OS
  - 1978年にApple社から提供された、GUIを持つ最初のOS
  - マウス操作やアイコンの概念を初めて導入
  - 現在はMac OS X

- Microsoft Windows
  - 1985年にMicrosoft社から初めて提供されたGUIを持つOS
    - 初期のWindows(Windows3.0, Windows3.1)はMS-DOSの拡張
    - Windows95で現在の原型
    - 現在のWindow 8までバージョンアップをして様々な機能を追加
    - サーバ用OSも提供

現在のOS[2](p. 75)

- UNIX
  - 歴史
    - 1969年にAT&T(アメリカの電話会社)から提供
    - 1970年から1980年初期まで大学や研究所などで利用
  - 現在はサーバとして多く利用
  - マルチユーザ環境
  - Solaris, AIX, SUN OSなど、様々な種類が存在(有料のものもフリーのものも存在)

現在のOS[3](p. 75)

- Linux
  - 1991年にリーナス・トーパスル(ヘルシンキ大学の学生)が開発
  - UNIX的なOS(使い方などがほぼUNIXと同じ)
  - オープンソースのフリーなOS
  - PCでも大型機でも利用(サーバとしても利用)

マルチユーザ環境

- 1つのコンピュータを複数の人が同時に利用可能
  - ネットワークを使って、あるコンピュータAから別のコンピュータBにログインすることができる
    - AにはなくてBにあるソフトウェアを利用したい場合(Aの前に別の人が座っている場合など)
    - 遠くからBを操作する必要がある場合(コンピュータの管理など)