データ構造について

河野　能知

2019.05.31
研究室ミーティング

データ構造

- 何をするにも情報の管理は大事
  - いろいろなデータ、**データ間の関連**をどのように管理するか
  - どのように**記録を取って、情報を整理**するか

- コンピュータでデータを管理する、処理する
  - コンピュータはどのように情報を表現できるか
  - コンピュータで表現できるようにデータを整理する
  - またはデータを扱い易くなるようにプログラムを組む

- 課題への取り組み方、結果の見せ方
  - 情報を整理することで、何をどのように調べるか明確になる
  - 大事なことを伝えるためにも情報を整理して見せる必要がある

どんなデータがあるか

- 天気の記録
  - 日付、気温、湿度、…
- 個人の予定表
  - 予定開始日時、終了日時、内容、他の参加者、…
  - 複数の人の予定で、人物間の関連も知りたい
- 物理学科に所属する人のリスト
  - 氏名、読み方、学年、出身校、履修科目、成績、進路、就職先、…
- 実験データ
  - 条件をいろいろ変えてデータを取る
  - 多次元

東京メトロの例

問題： 地下鉄の運行は適切にされているか？

「適切」とは何か

乗客と運行会社、双方が最大限満足できる条件

	便益	費用
運行本数を増やす	乗客の利便性が高まる。	増やし過ぎると、無駄が増える。 コストも増える。
運行本数を減らす	乗客の利便性が下がる。 利用者が減って収入が減る。 行政からも指導を受ける。	コストが減る。

- 利便性を数値化する。（混雑度合い、待ち時間、目的地までの乗り換え回数）
- 現状を把握する
  - 客観的なデータを集める
  - データを基に何をすべきか決定する
- 最後は関係する人達が納得するように、決定に至った丁寧に説明する

地下鉄の運行状況、乗客の満足度を把握する

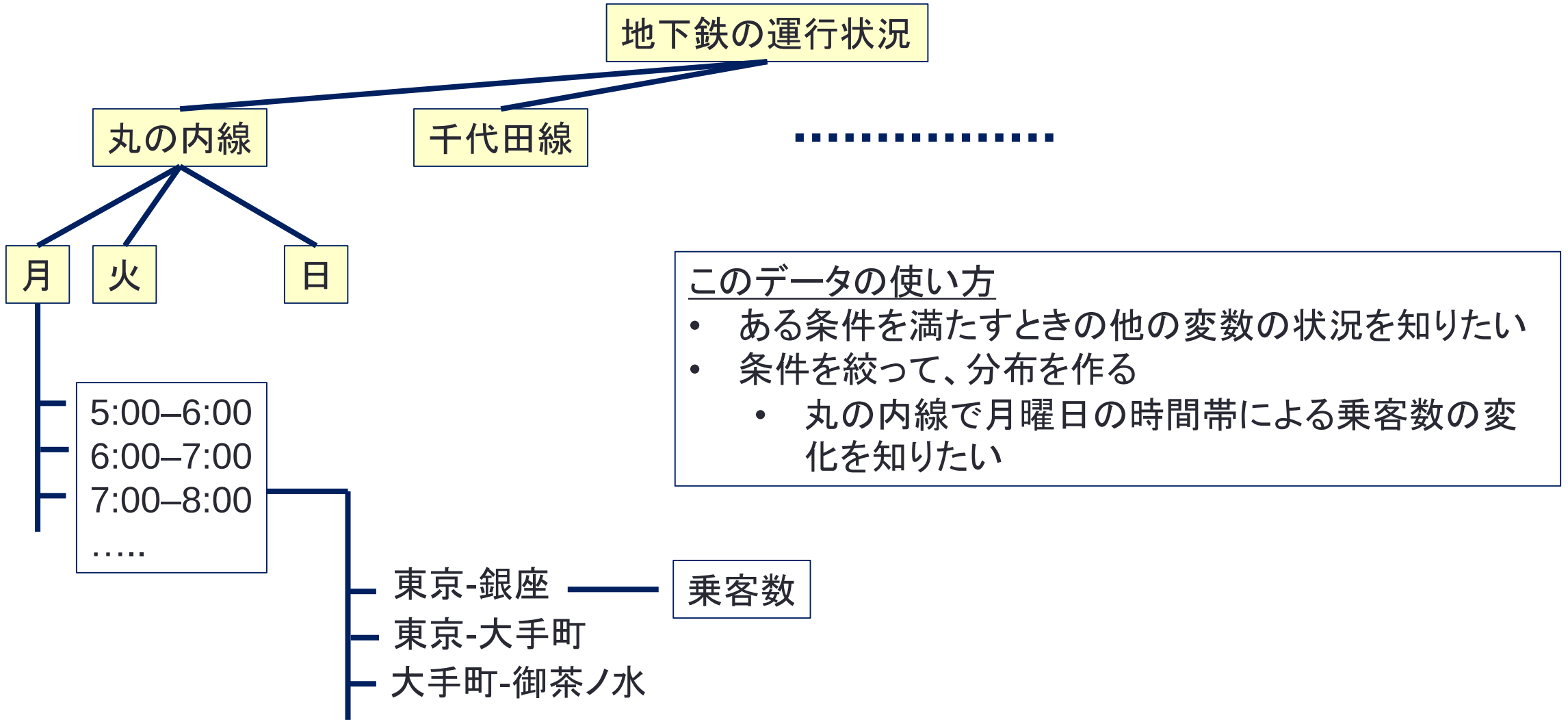
- 測定する量
  - 各車両の混雑状況
    - 路線、曜日、時間帯ごとに測る
    - 天候
    - 大きなイベントの開催状況
  - …..

月	火	…..
丸の内線		
千代田線		
半蔵門線		
銀座線		
日比谷線		
…..		

さらに、
- 時間帯
- どの駅間か
- 天候
- イベント開催状況
- …..

このような多変数データをどう整理するか

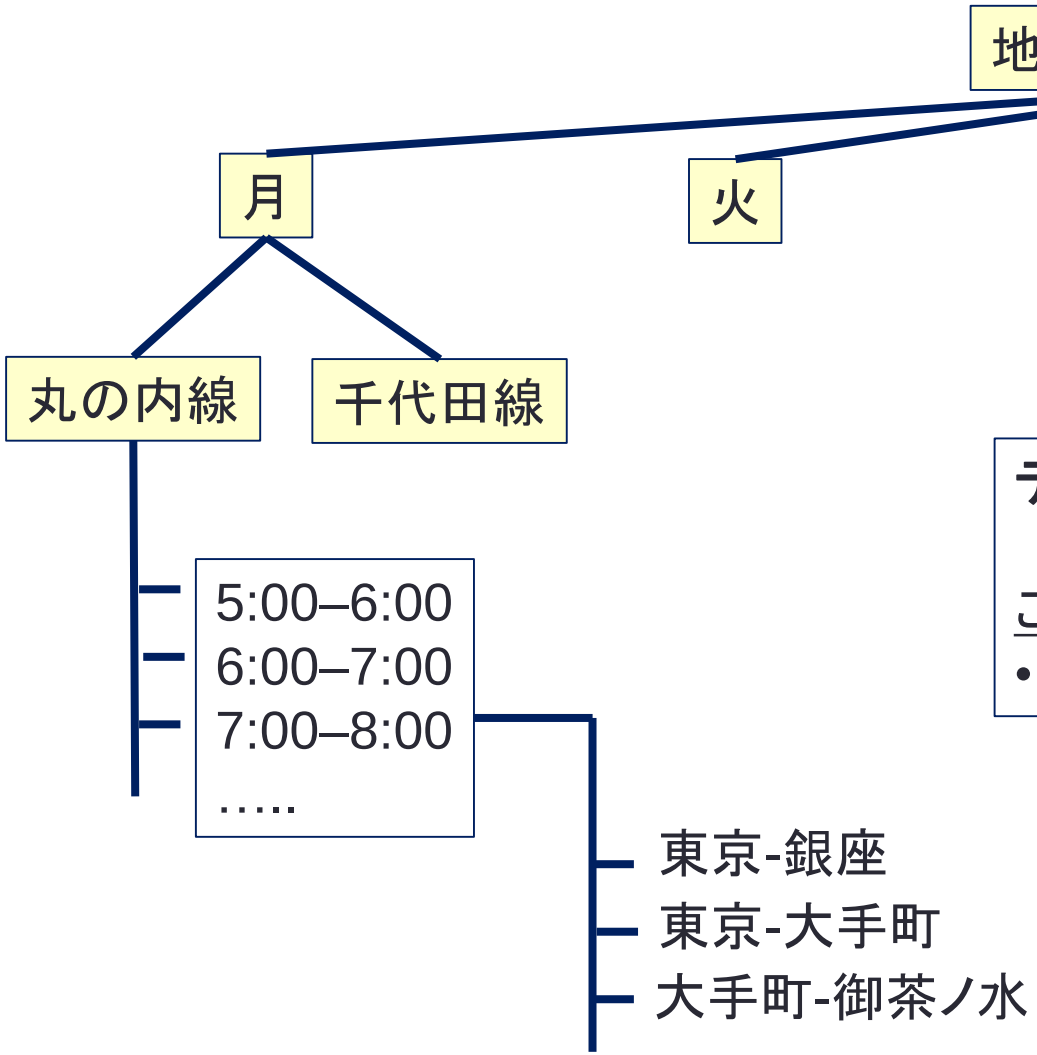
データの整理(1)



このデータの使い方

- ある条件を満たすときの他の変数の状況を知りたい
- 条件を絞って、分布を作る
  - 丸の内線で月曜日の時間帯による乗客数の変化を知りたい

データの整理(2)



データの整理の仕方は一通りとは限らない

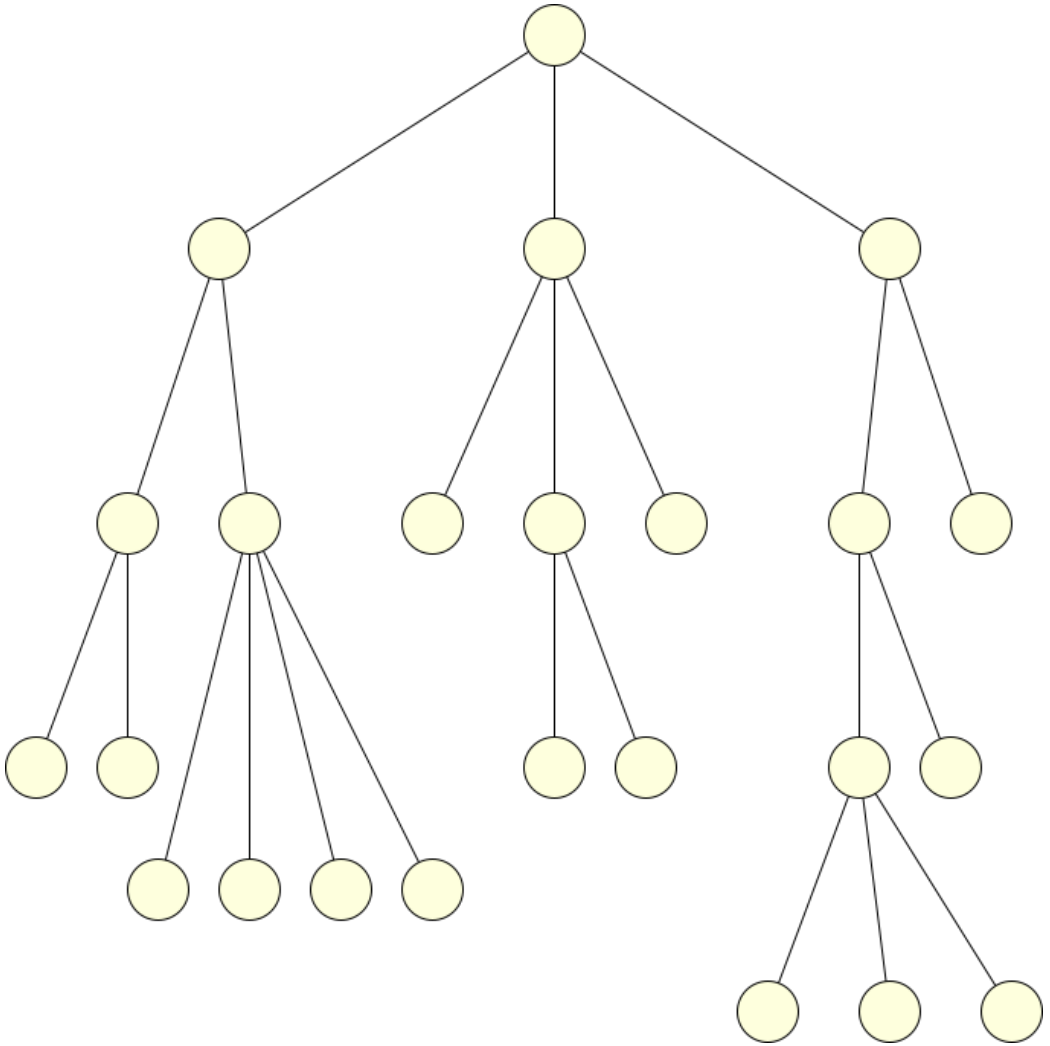
<u>このデータの使い方</u>

- 時間帯ごとに、駅間の乗客数の地図上に示したい

データ分析

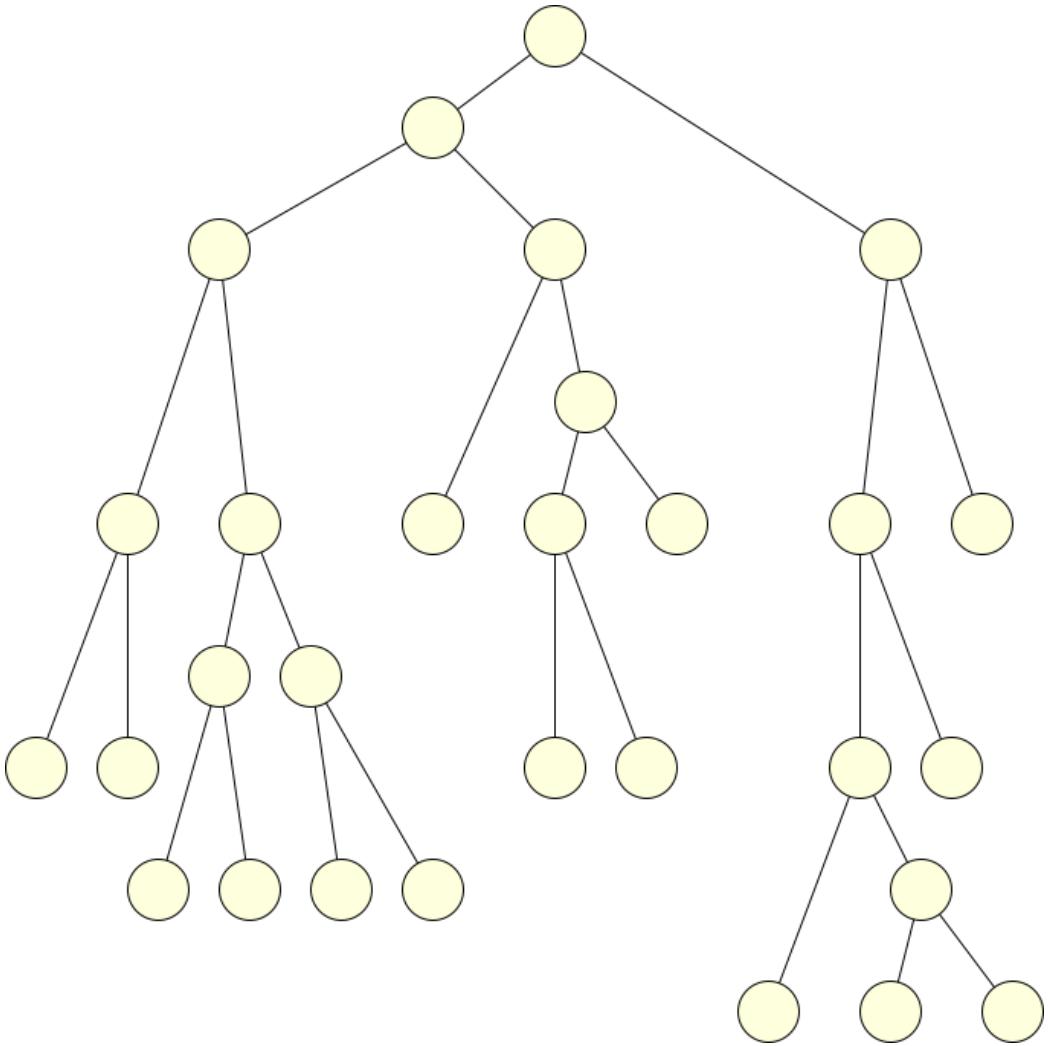
- ある条件を満たすとき、他の変数の状況を知りたい
  - 最も乗客数が多い（少ない）路線、時間帯はどれか
- ある条件を満たすとき、他の変数の分布を作りたい
  - 一日の乗客数の変化を曜日ごとに知りたい
  - 路線による違いはあるかどうか
- データの特定と検索
  - データ構造によって、データへのアクセスのし易さは異なる
  - 利用法に合ったデータの整理の仕方を採用するべきである

木構造



- データを階層的に関連付けたもの
  - ノードと枝で表現
  - 親子関係（親は一つ、子は複数可）
  - 子のないノードを葉とよぶ
- いろいろなデータをこのような構造で表現できる
  - ディレクトリ構造
  - 文書の構成（章、節、段落、…）
  - 組織の階層
  - 辞書
- 木構造でないもの
  - 家系図（複数の親）
  - 環状のグラフになるもの

二分木（Binary Tree）



- データを階層的に関連付けたもの
  - ノードと辺で表現
  - 親子関係（親は一つ、子は0 or 2のみ）
- どんな木でもノードを追加すれば二分木で表現できる
- 枝ノードに明確な順序付けが可能
  - 要素の大小関係を定義できる場合（>）
  - 辞書

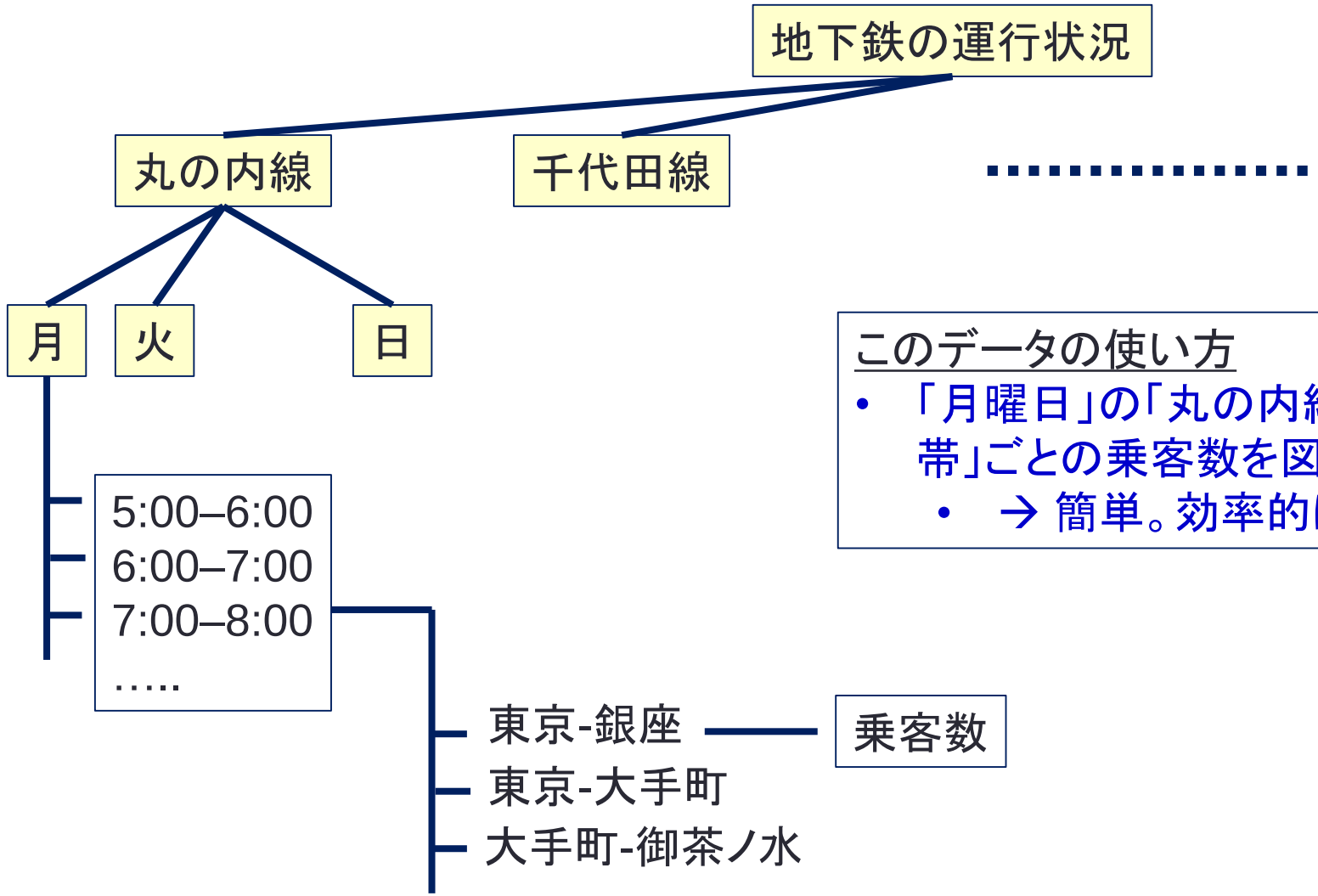
木構造と表

路線名	曜日	時間帯	出発駅	到着駅	近隣のイベント …..	乗客数
丸の内線	月	6:00-7:00	東京	銀座	なし	100
丸の内線	月	7:00-8:00	東京	大手町	なし	120

重複

- 木構造のデータを2次元の表で表すことも可能である。（平坦化）
- 末端の葉に対して、その上にあるノードの値を保管する列を用意すればよい。
  - 但し、データの重複が増える
  - 一方、表は構造が単純なので、検索性は全ての変数に対して同等である

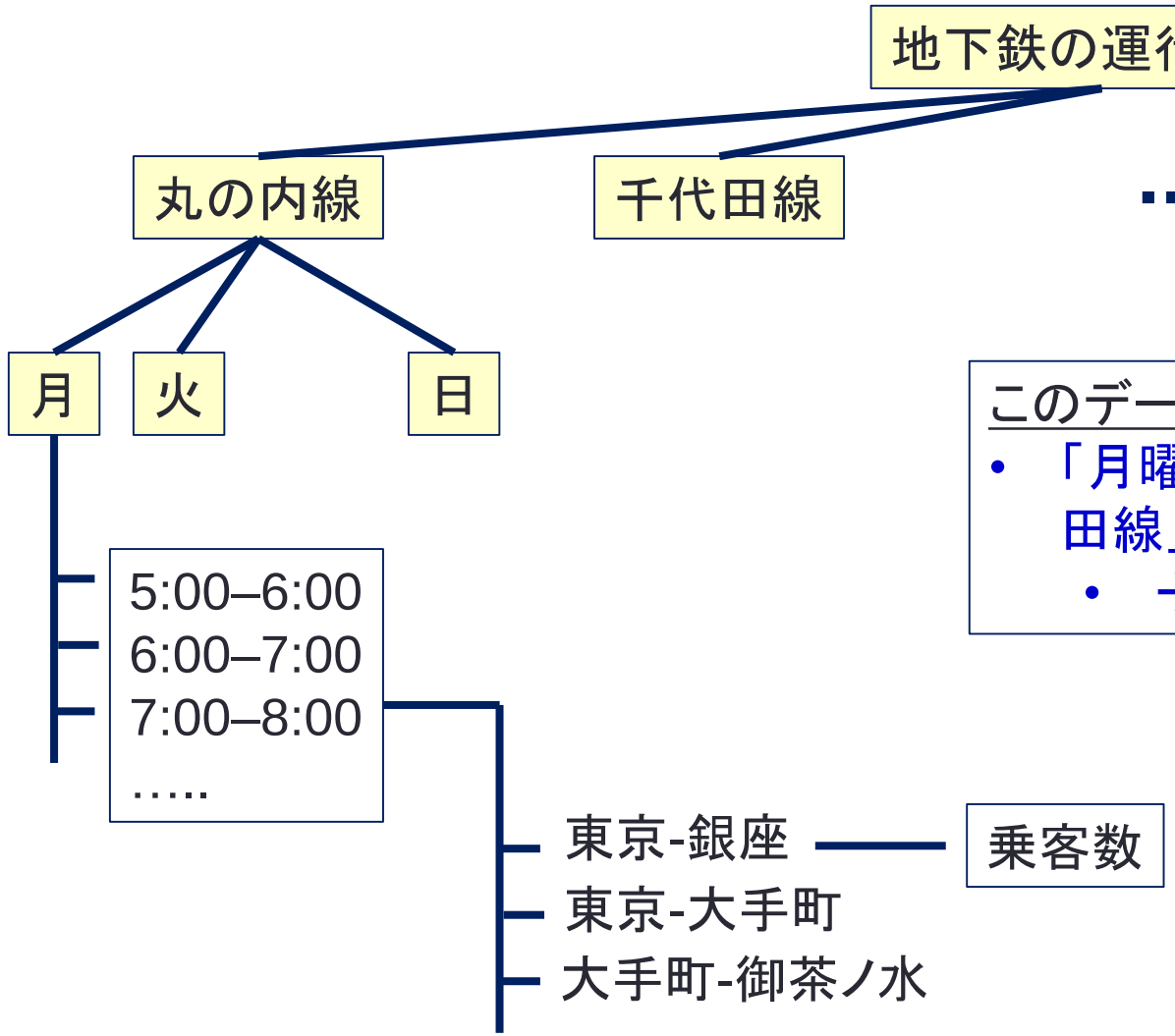
データのスキャン（1）



このデータの使い方

- 「月曜日」の「丸の内線」の「東京-銀座間」の「時間帯」ごとの乗客数を図示したい
  - → 簡単。効率的にデータを取り出せる

データのスキャン(2)



このデータの使い方

- 「月曜日」の「7:00-8:00」における「丸の内線」と「千代田線」の全乗客数を比較したい
  - → データが散らばっていて分析が複雑

木と表

- 大規模データ
  - 例で挙げた「地下鉄の乗客数を調べる」ことで得られるデータ
    - それぞれの「地点」での観測にはいろいろな属性がつく。（路線、曜日、時間帯、駅、乗客数、…）
  - 一般に、「複数の属性をもつ同種のデータ」を大量に扱うことが多い
- 木
  - いろいろな属性に対して優先順位を設定して、優先度の高い属性からデータを分類して整理
    - どのように優先順位をつけるかが考えどころ
  - 必要なデータの探索と分類の順序と合っていれば効率的に検索が可能
  - 必要なデータと分類の順序が違うと、検索が非効率になる
  - 結果の発表には、どのようにデータを分析したかを明確にするため木構造の考え方は重要
- 表
  - 2次元の表なので、構造が単純
  - 但し、データの重複が出る
  - データ選択の条件は各列の値を指定することでできる。全ての列が対等

プログラミング言語におけるデータ構造

- 複雑なデータ構造をどのように表現するか
- 同じ種類のデータを大量に扱いたい
  - → 配列
- 異なる種類のデータをまとめて扱いたい
  - → クラス
- いろいろなデータ構造
  - vector, list, set, map