

# プロットの作り方 v16

---

塩谷 亮太

# 3点プロットのチェック・リスト

3点プロットを作ったら、以下が満たされているかを確認する：

（「形式に関するチェック」については、他のタイプのプロットでも同様に確認する）

## ■ 内容に関するチェック：

1. 背景，課題，提案の3項目から構成されている
2. 背景は，課題と提案に向けて話題を絞り込んでいる
3. 課題は，「～が悪い」「～が遅い」などの問題を直接示す文を含んでいる
4. 提案は，上記の問題が「どのように」「なぜ」解決されるのかを示す文を含んでいる

## ■ 形式に関するチェック：

5. 各箇条書きは複文を含んではならない
6. 1行を越えるような長い修飾節を含んだ文を含んではならない
7. 4つ以上の項目を並列に並べてはいけない
8. 箇条書きの親子関係で説明されている「階段」を作ってはいけない

はじめに

---

# はじめに

- プロットとは：
  - ◇ 文章の論理構造を箇条書きの形でまとめたもの
  - ◇ 話の筋をまとめたもの，と考えてもよい
- 文章や発表スライドを書く前に，まずプロットを作る必要がある
  - ◇ これはいわば文章やスライドの設計図にあたるもの
  - ◇ いきなり文章を前から順に書き始めてはいけない
    - 論理的な構造をきちんと設計したあとで書き始める

# なぜプロットを作るのか？

- 話の筋を整理するため
  - ◇ 何が背景で、何が課題で、何をどう解決したのかを明確にする
- その筋に収束するよう文章を書くと、主張を明確に示すことが出来る
  - ◇ そうしないと、「言いたいことがなんとなく適当に並べられた良くわからないもの」が出来上がる
  - ◇ 設計図なしで建物を建てるとヒドい事になるのと同じ
- 論文の章構成レベルの設計をしているとも言える
  - ◇ これを先にやっておかないと、後から大きな手戻りが発生する
  - ◇ 文章の修正が細かい手直しではすまなくなる

# プロットにはいろいろなタイプがある

論文や発表スライドの作成段階に応じて、これらを作る

## 1. 3点プロット

- ◇ 話全体を3つの項目にまとめた形のプロット
- ◇ イン트로/全体プロットと比べると抽象的な内容になる

## 2. イントロプロット

- ◇ 論文やスライドのイントロのプロット

## 3. 全体プロット

- ◇ 論文やスライド全体のプロット（章立て）

## 4. 目標規定文

- ◇ 論文やスライドの内容を1文で表したもの
- ◇ ここではプロット的一种と考えている

# 3点プロットとは

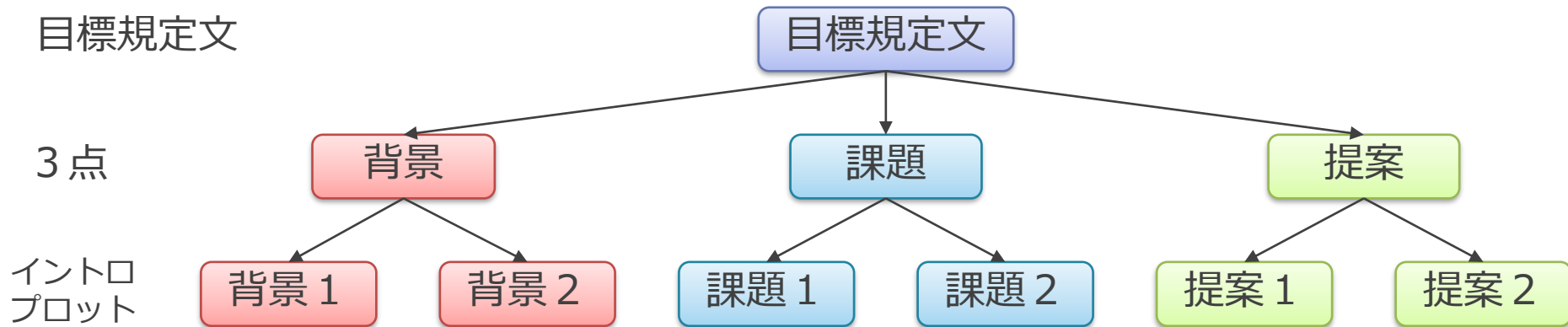
- 3点プロットはいわば「プロットのプロット」
  - ◇ 「背景→課題→提案」の3要素が何なのかをまとめたもの
  - ◇ 各要素が何なのかをはっきりさせ、その流れを明確にする
  - ◇ これを膨らませて全体プロットなどを作る

# まず3点プロットから作り始める

- イントロプロットや全体プロットをいきなり作るのは難しい
  - ◇ 自由度が高すぎて、いきなりまとめるのが難しい
- 目標規定文を最初に作るのも難しい
  - ◇ 本当に大事な事だけを1つの文に集約/圧縮する必要がある
  - ◇ しかし、何が真に大事なのかは最初はわからない
- 3点プロットが規模的に最初に手をつけるのにちょうどよい
  - ◇ 規模が小さくかつ形式が決まっているので、考えやすい
  - ◇ 典型的にはスライド1枚程度にまとめる
  - ◇ 短いので、まず取っ掛かりとして始めやすい



# 詳細度と論理構造



- ツリーの上下は基本的に説明の詳細度に対応している
  - ◇ 上の階層は下の階層の要約になっている
  - ◇ 下の階層は上の階層をより詳しく述べている
- 3点プロットから初めて,
  - ◇ 上に登る (=概要にまとめる) ことや,
  - ◇ 下に降りる (=詳細を肉付けする) していく とよい
- これらの親子関係が厳密には成立しない場合もあるので注意  
(全体プロットの節などで説明)

# 全体の目次

1. プロットの作り方
  1. 3点プロット
  2. 目標規定文
  3. イントロプロット
  4. 全体プロット
2. プロットから文章へ

# 3点プロット

---

# 3点プロットの目次

1. 3点プロットとは
2. 内容のまとめかた
3. 箇条書きの作り方
4. 余談：なぜ箇条書きにまとめるのか？

# 3点プロット： 背景，課題，提案の3点で話の筋をまとめる

## 1. 背景：主張全体の背景や問題を説明する

- ◇ 課題と提案に共通する背景や，それらが共通して取り組んでいる問題
- ◇ 一般的な話題から始めて，課題や提案へ向けて話題を絞り込む

## 2. 課題：解決しようとしている課題を説明する

- ◇ 既存手法の問題点
- ◇ 既存手法がない場合は，背景の中の着目する問題を掘り下げる

## 3. 提案：課題であげられた問題を解決する提案手法を説明する

- ◇ 課題に対する洞察や観察
- ◇ キーとなるアイデア
- ◇ なぜ & どのように課題を解決できるのか

# 例 1 : セットアソシアティブ・キャッシュ = 既存手法があるパターン

## ■ 背景：キャッシュ

- ◇ 目的：プロセッサとメイン・メモリ間の速度差の解消
- ◇ 構造：高速/小容量なメモリであり，メイン・メモリの一部を保持

## ■ 課題：既存のキャッシュは性能 or 複雑さに問題がある

- ◇ ダイレクト・マップ：単純だが，競合によるヒット率低下が大きい
- ◇ フルアソシアティブ：ヒット率は高いが，大量の比較器が必要

## ■ 提案：セットアソシアティブ・キャッシュ

- ◇ 手法：複数のラインを同時に保持するセットを用いる
- ◇ 効果：
  - ダイレクトマップと比べて競合にある程度耐性があるため，ヒット率が高い
  - フルアソシアティブに比べて比較器の数は大幅に少なく単純

## 例 2 : 小泉くんの DATE = 既存手法があるパターン

- 背景 : 早いプリフェッチによるレイテンシの隠蔽
  - ◇ プリフェッチ : キャッシュにデータを先読みしておく技術
  - ◇ 多くの既存研究では十分に早くプリフェッチすることを重視
    - メモリ・アクセスのレイテンシを有効に隠蔽するため
    - 通常はなるべく遠い未来のアドレスを予測してプリフェッチ
- 課題 : 早すぎるプリフェッチ
  - ◇ 早すぎると, 使用される前にキャッシュから追い出される
  - ◇ これにより性能向上の機会が大きく失われている
- 提案 : プリフェッチを遅らせる
  - ◇ データが参照されるタイミングまであえて遅らせる

# 例 3 : 小田喜くんの輪講の例 = 既存手法があるパターン

(輪講なので具体的なアイデアがまだない事に注意)

## ■ 背景 : SIMT アーキテクチャにおける冗長な演算

- ◇ SIMT(D) では基本的には複数のデータに対して同じ演算を行う
- ◇ しかし SIMT では全く同じ冗長な演算を複数のレーンで行っている場合があります無駄

## ■ 課題 : スカラ化とその問題

- ◇ 冗長な演算を 1 つの演算にまとめるスカラ化が提案されている
- ◇ しかし, 従来のスカラ化では制約があり効果的に演算をまとめられない

## ■ 提案 : スカラ化の改良

- ◇ Temporal SIMT と動的なスカラ化の組み合わせにより実現



# 例 4 : 出川くんの ICCD = 既存手法がないパターン

- 背景：命令キャッシュ・ミス数を使った性能の見積もり
  - ◇ 従来，命令キャッシュに関わる研究ではミス数が主要な評価項目だった
  - ◇ 理由：ミス数が減ると基本的には実行時間が短くなるため
- 課題：精度とシミュレーション時間
  - ◇ 動機：
    - 現代の複雑化したプロセッサではミス数と実行時間が直接相関しない
    - 精度よい性能見積もりのためにはプロセッサ全体のシミュレーションが必要
  - ◇ 問題：しかしそのようなシミュレーションには長い時間かかる
- 提案：命令キャッシュ・ミス数に代わる新たな指針
  - ◇ 手法：新たな指針と，その指針を使った高速な性能見積もり
  - ◇ 効果：2桁短い時間でシミュレーションとほぼ同じ精度の性能見積もりを実現

# 例 5 : 木村さんの輪講 = 既存手法がないパターン

## ■ 背景 : ベクトル命令

- ◇ 単一の命令で可変長の複数データを処理する命令の方式
- ◇ データ並列性のある処理を対象
- ◇ 例 : RISC-V ベクトル拡張など

## ■ 課題 : ベクトル命令の実装コスト

- ◇ ベクトル命令では 1 つの命令が多数のアクセスを発生させる
- ◇ 従来の作り方で out-of-order プロセッサ上に実装すると、複雑さが爆発する

## ■ 提案 : ベクトル命令の実装の複雑さを下げる

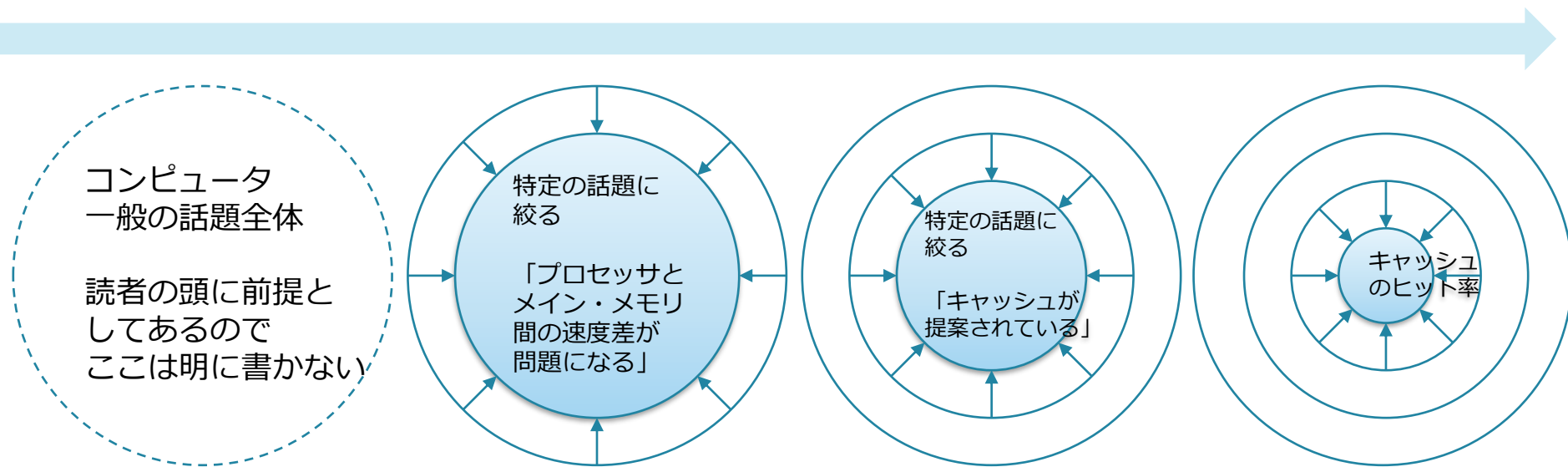
- ◇ 部分的な in-order 実行の導入による複雑さの緩和
- ◇ in-order/out-of-order 部の軽量な同期方法の提案

# 背景について

- 「背景」はプロット全体の議論の場を設定する
  - ◇ まず、何についての話題なのか
  - ◇ その話題において何が主に問題となるのかや、目的は何なのか
  - ◇ 「課題」や「提案」で話す内容に話題を絞っていく
- たとえば、前述の例の課題の場合：
  - ◇ 「キャッシュ」
  - ◇ 「早いプリフェッチによるレイテンシの隠蔽」
  - ◇ 「SIMT アーキテクチャにおける冗長な演算」
  - ◇ 「命令キャッシュ・ミス数を使った性能の見積もり」
  - ◇ 「ベクトル命令」

# 背景の話題の絞り方

- 一般的な事項から話題を絞っていく
  - ◇ 「課題」や「提案」の話題がちょうど含まれるところまで絞る
  - ◇ 絞りきったところで,
    - それを一言にまとめたものを「背景」のトップレベルに書く
    - その下に絞っていく過程をぶら下げる



# 背景を絞る例（小泉くんの DATE の背景）

- 背景：早いプリフェッチによるレイテンシの隠蔽
  1. メイン・メモリのアクセス・レイテンシが大きな問題に
  2. プリフェッチ：キャッシュにデータを先読みしておく技術
  3. 多くの既存研究では十分に早くプリフェッチすることを重視
    - メモリ・アクセスのレイテンシを有効に隠蔽するため
    - 通常はなるべく遠い未来のアドレスを予測してプリフェッチ
- 上記の背景は、以下のようにして話題を絞り込んでいる
  - ◇ 「1. メモリのレイテンシが問題」 →
  - ◇ 「2. プリフェッチによる解決」（レイテンシ問題の解決法の1つ） →
  - ◇ 「3. 早いプリフェッチの重視」（プリフェッチの性質の1つ）
- このようにして、その後の「課題：早すぎるプリフェッチ」と「提案：プリフェッチを遅らせる」に繋げている

# 課題について

- 「課題」は、良くない事を示すネガティブな語句や文を必ず含む
  - ◇ 「～が悪い」「～できない」など
  - ◇ これにより、なにが問題であるのかを明示する
- たとえば、前述の例の課題の場合：
  - ◇ 「従来のスカラ化では制約があり効果的にまとめられない」
  - ◇ 「性能向上の機会が大きく失われている」
  - ◇ 「複雑さが爆発する」
  - ◇ 「シミュレーションには長い時間かかる」

# 項目間の関係

- 「課題」に書く内容は、「背景」で提示した問題に対応させる
  1. 既存手法がある場合：
    - 「背景」で提示した問題を解決する
    - （基本的にはこの形になることが多い）
  2. 既存手法がない場合：
    - 「背景」の特定の問題に着目して掘り下げる
- 「提案」では、「背景」と「課題」で提示した問題に対応させる
  - ◇ 基本的には「課題」で提示した問題を解決する方法を書く
  - ◇ そうすれば、普通は自然と「背景」の問題にも対応する

# 応用：4点プロット

## ■ 洞察を加えた4点プロットの形でもよい

◇ 背景

◇ 課題

◇ 洞察

□ 課題に対する新しい観察結果や、課題の核心の新しい解釈など

◇ 提案

## ■ 洞察は課題や提案の下にぶら下がることもある

◇ 小泉くんの例での「早すぎるプリフェッチ」は洞察でもある



# 3点プロットの目次

1. 3点プロットとは
2. 内容のまとめかた
3. 箇条書きの作り方
4. 余談：なぜ箇条書きにまとめるのか？

# とりあえず、この形にまとめる事を目指す

「～」 となっているところは、一つの文ないしは名詞による説明

## 1. 背景：「～」

- ◇ 「～」 = 背景を説明する 1 つの文
- ◇ 「～」 …

## 2. 課題：「～」

- ◇ 「～」
- ◇ 「～」

## 3. 提案：「～」

- ◇ 「～」
- ◇ 「～」

# 作り方の例

- 作り方の例

1. ボトムアップな作り方
2. 課題から掘り下げる作り方

- やりやすいようにやれば良いし, 上記を組み合わせても良い

- ◇ 他にも色々なやり方があると思う

# ボトムアップな方法

1. まずは思いつく関連しそうな項目をたくさん書き出してみる

- ◇ 名詞や短文の形にして並べてみる

2. 各項目を整理

1. 関係する項目同士をくくって親子にまとめる

- それらを一言で表した、まとめの短文（親）を作る
- 親の下に、それらを子項目としてインデントして置く

2. 内容が冗長なものをマージしたり削除する

3. ある程度まとまったら、それらを背景，課題，提案に分類する

- ◇ このとき、「項目間の関係」のページで説明した関連を意識する
- ◇ うまく分類して関係を説明できない時は、本質的に関係ない可能性がある

# 課題から掘り下げる方法

- まず「課題」は何であるのかから考える
  - ◇ なにが問題なのかを端的にまとめる
  - ◇ 「～が悪い」「～が遅い」「～の効率がよくない」などの形の1文に出来るとよい
- 次にそれを「提案」がどのように解決しているのか考える
  - ◇ 上でまとめた問題が、「どのように」「なぜ」解決されているのかをまとめる
- それら「課題」と「提案」に話題を絞り込んでいくような「背景」を考える

# 箇条書きの作り方

---

# 3点プロットの目次

1. 3点プロットとは
2. 内容のまとめかた
3. 箇条書きの作り方
  1. 文を短くする
  2. 属性を使った親子関係の作り方
  3. インデントにぶらさげる項目数
  4. 親子関係における「階段」
4. 余談：なぜ箇条書きにまとめるのか？

# 文を短くする

## ■ 複文を使うのは原則禁止

- ◇ 複数の文を繋げて 1 つの文にしてしまうと、関係が良くわからなくなりがち
- ◇ 最初は単文のみで作る
  - どうしても複文を入れる場合は、1 行に収まる長さまで
  - 3 文以上からなる複文は常に禁止

## ■ 同様の理由により、長い修飾節を持った文も使わない

- ◇ 長くなってしまう場合は、インデントをしてぶら下げると良い



# 箇条書きの親子関係の作り方

- インデントされた子要素の部分と、その親の関係
  - ◇ 子項目は、その親項目のなんらかの詳細を説明する
  - ◇ 親項目は、その子項目をまとめた内容となる
- 項目の「属性」や「話題」に従ってくくっていくと良い

# 属性による親子関係の確認


- 各項目の先頭に一言にまとめた属性をつけて作ると確認しやすい
  - ◇ 子から見て親の何であるのかを属性としてつける
  - ◇ 「問題：」「理由：」「結果：」「目的：」「例：」「詳細：」など
- 属性をつけた例：
  - ◇ 背景：ベクトル命令
    - 詳細：単一の命令で可変長の複数データを処理する命令の方式
    - 目的：データ並列性のある処理を対象
    - 例：RISC-V ベクトル拡張などの形で実装されている
- 属性は、プログラミング言語における型の概念に似ている
  - ◇ 構造体や配列と似たような構造化の考え方ができる
  - ◇ 「違う型のものを配列に入れてはいけない」のような概念が応用できる

# 共通の属性をくくる

- 同じレベルにある複数の同じ属性の項目は、くくって1つにする
  - ◇ 複数の同じ属性（型）のものが別の属性（型）と同じレベルに並んではいけない
- たとえば、2つの「例：」が「詳細：」と同じレベルにあるような以下の場合：
  - ◇ 「例：」をくくって1つの配列にする
  - ◇ 背景：ベクトル命令
    - **詳細**：単一の命令で可変長の複数データを処理する命令の方式
    - **例**：RISC-V ベクトル拡張
    - **例**：NEC SX
  - ◇ 背景：ベクトル命令
    - **詳細**：単一の命令で可変長の複数データを処理する命令の方式
    - **例**：
      - \* RISC-V ベクトル拡張
      - \* NEC SX



# 共通の話題をくくる

- 同じレベルにある複数の同じ話題の項目も、くくって1つにする
    - ◇ 属性とは直行している
  - たとえば、「ベクトル命令」について話している以下のような場合：
    - ◇ それらを「ベクトル命令」でくくる
    - ◇ 詳細：ベクトル命令とは単一の命令で可変長の複数データを処理する命令の方式
    - ◇ 例：ベクトル命令には以下のような実装の例がある
      - RISC-V ベクトル拡張
      - NEC SX
- 
- ◇ ベクトル命令：
    - 詳細：単一の命令で可変長の複数データを処理する命令の方式
    - 例：実装の例：
      - \* RISC-V ベクトル拡張
      - \* NEC SX

# インデントにぶらさげる項目数

- 一つの項目にぶら下げる項目は3つまで
  - ◇ 4つ以上の項目が同じレベルに並んでいる場合は、それらをインデントにまとめる
  - ◇ 3つより多いと関係が曖昧になるし、理解しづらい

# 箇条書きの親子関係における「階段」

- $A \rightarrow B \rightarrow C$  のような演繹の関係を下のような「階段」のような箇条書きに  
してしまいがちだが、これは良くない

- ◇ (親子関係は基本的に 概要 $\leftrightarrow$ 詳細 を表すもの)

- ◇ A : ~

- B : ~

- ✱ C : ~

- このような場合は  $A \rightarrow B \rightarrow C$  の主張を一言にまとめたものを (X) を作り、  
その下にぶらさげる

- ◇ X :

- A : ~

- B : ~

- C : ~

# 演繹の関係にある要素の書き換えの例

## A→B→C を X の下に展開

各インデントレベルに 1 つだけ項目がある階段が出来てしまっている

- A : 人間の脳の一時記憶の大きさには限りがある
  - ◇ B : このため, 人間は 5 ~ 7 個以上の事柄を一度に把握できない
    - C : したがって, 余裕を持って 3 個程度以内にするのがよい

X に全体をまとめる一言を入れて, その下に並列にぶら下げると良い

- X : 1 項目にぶら下げる項目の数は少なめにする
  - ◇ A : 人間の脳の一時記憶の大きさには限りがある
  - ◇ B : このため, 人間は 5 ~ 7 個以上の事柄を一度に把握できない
  - ◇ C : したがって, 余裕を持って 3 個程度以内にするのがよい

# 3点プロットの目次

1. 3点プロットとは
2. 内容のまとめかた
3. 箇条書きの作り方
4. 余談：なぜ箇条書きにまとめるのか？



# 余談：プロットの作成時に なぜ親子関係のある箇条書き（階層構造）にまとめるのか？

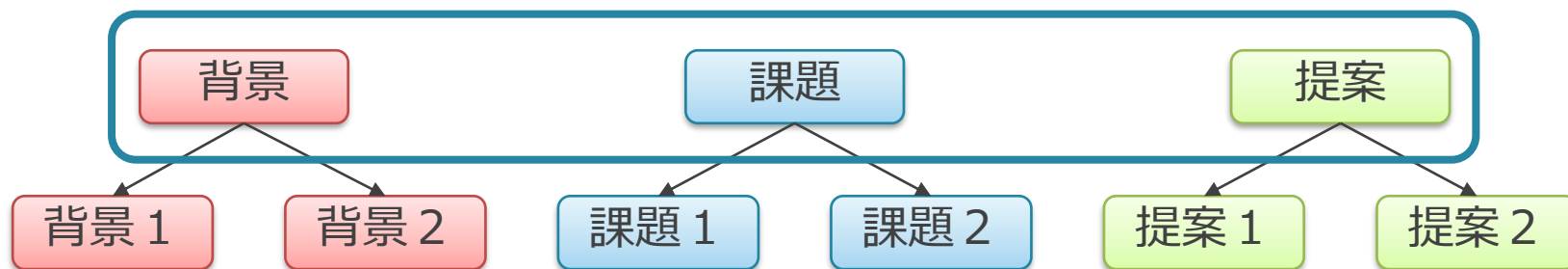
- すべての項目がフラットに並んでいると内容の把握が難しい
  - ◇ 人間の頭が一度に扱える量には限界がある
  - ◇ 5個ぐらいからは並列に並んでいると厳しくなってくる
- 階層化して一度に考えることの数減らす
  - ◇ 関係ある項目ごとに要約にまとめて階層化
  - ◇ 階層化すると、一度に考えることの数減る
- 一度に考える話題の数が小さいままに、以下が両立できる：
  - ◇ 話の大筋をつかむ
  - ◇ 各部分の詳細を理解する

# 一度に考える必要がある話題の数

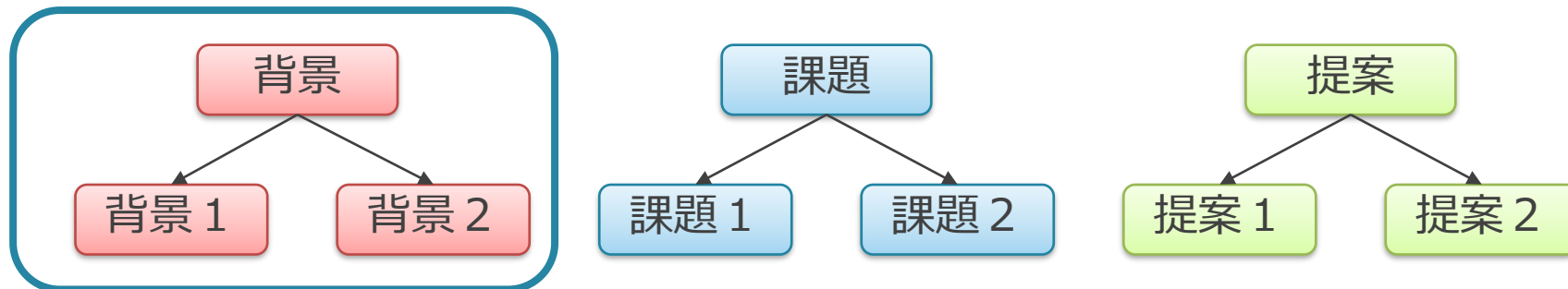
- フラットに並んでいる場合：全体を理解する=話題数 6



- 階層化されている場合：全体の話の大筋をつかむ=話題数 3



- 階層化されている場合：「背景」部分の詳細を理解する=話題数 3



# 階層化を意識することの重要さ

- 論理の階層化は、複雑な事象を考えるための必須スキル
  - ◇ 思考の規模をスケールさせる事ができる
  - ◇ 一定以上の経験をもつ人は、普段の思考からこの階層化を行っていると思って良い
- なので、まともな論文や説明は階層化された形で書かれている
- 逆に、この階層化がなされていない説明を読むことは苦痛である
  - ◇ 誤字脱字文法ミスだらけの文章を読まされるのと同等以上のきつさがある
  - ◇ 査読者などの読み手はそう感じると想定してほしい

## ■ バーバラ ミント：

「考える技術・書く技術—問題解決力を伸ばすピラミッド原則」  
(原題：The Pyramid Principle: Logic in Writing and Thinking)

- ◇ 思考や文章を書く際の論理の階層化について書かれている
- ◇ マッキンゼー社内でライティング指導をしていた方が書いている
- ◇ このページまで読んできた人は、「ピラミッド」が何を意味するのかは察しがつくのでは

## ■ 塩谷の感想（読んだのは10年以上前だが）

- ◇ 翻訳版はかなり読みづらいし、内容もかなり冗長さを感じる
- ◇ でも大事なことが書かれていると思う
- ◇ 後述の「理科系の作文技術」と、もっとも大事な部分では同じ事を言っていると思う

# 目標規定文

---

# 目標規定文

## ■ 目標規定文

- ◇ その文章の主張を 1 つの文の形にまとめたもの
  - 目標規定文は「理科系の作文技術」より
- ◇ もっとも短い形のプロットとも言える

## ■ 「理科系の作文技術」の説明

- ◇ 「自分は何を目標としてその文章を書くのか、そこで何を主張しようとするのかを熟考して、それを一つの文にまとめて書いてみることを勧める」
- ◇ 「主題に関してあることを主張し、または否定しようとする意思を明示した文」（コーベットによる thesis の説明）

# 目標規定文

- 目標からトップダウンに構成を作る
  - ◇ 目標規定文を作り，その目標に収束するように文章全体の構想を練る
  - ◇ この目標に繋がる内容のみを全体の構成に残す
    - 「関係はしているが，あってもなくても良い」みたいなものは入れてはいけない
- 主張全体を論理的なツリーとして表した際のルートにあたる
  - ◇ これをさらに短くまとめたものがタイトルになる

# 3点プロットと目標規定文の関係

## ■ 作り方：

- ◇ 3点プロットが出来たら，そこからさらに真に重要な項目を抽出
- ◇ それらを繋げて目標規定文にする



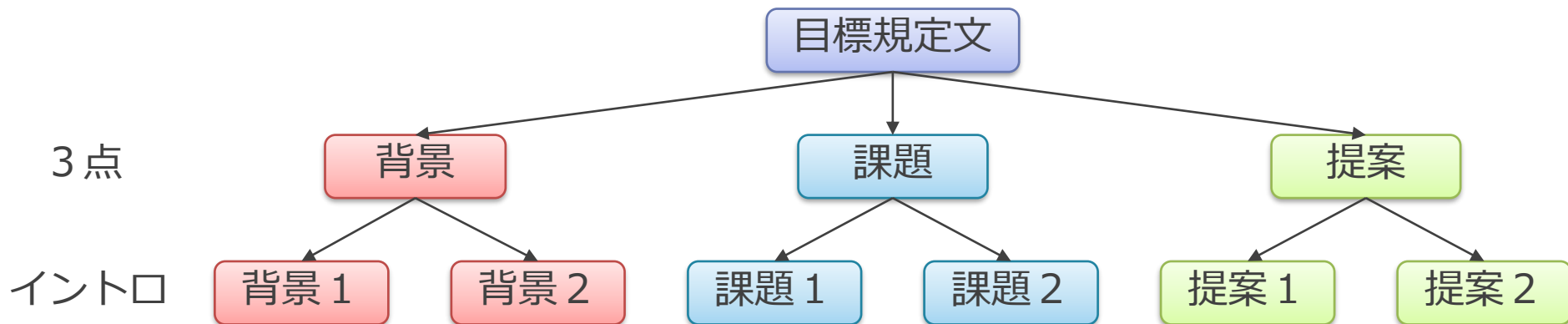
# イントロプロット

---

# イントロプロット

- イントロプロットはイントロを書く際に作る
  - ◇ 論文のイントロは典型的には6パラグラフ前後
- これに対応する6点程度の項目からなるプロットを作る
  - ◇ 各パラグラフで何を話すかをまとめる
  - ◇ 各項目は各パラグラフのトピック・センテンスの内容に対応する
  - ◇ 6点の項目だけを繋げて読んでも意味が通るようにする
- イントロプロットは3点プロットから派生させて作る
  - ◇ 基本的には3点プロットの各項目に、より詳細を肉付けしていく
  - ◇ ただし役割や目的の違いにより、作り方が異なる部分もある

# 詳細度と論理構造

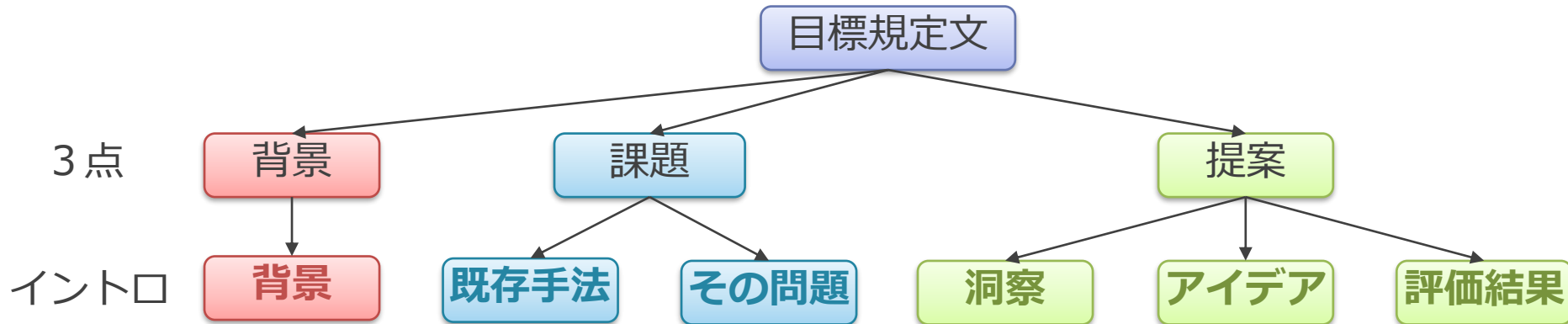


- ツリーの上下は説明の詳細度に対応している
  - ◇ 上の階層は下の階層の要約になっている
  - ◇ 下の階層は上の階層をより詳しく述べている
- 3点プロットから初めて、論理ツリーの関係が保たれているかを確認する

# イントロプロットと3点プロットの違い

- 3点プロットとは異なる部分がある
  1. 配分の違い
  2. イントロの役割に由来する違い
  3. プロットで確認する内容の違い

# イントロプロット時の配分



- イントロプロット時の配分には自由度がある
  - ◇ 3点のどこをどれだけ詳しく話すかは、論文ごとに異なる
- たとえば、
  - ◇ 背景は1つのままで、提案を増やすことが多い（上の図の例）
  - ◇ あまり一般的ではない話題の場合、背景が多めの配分になる事も
  - ◇ 逆に課題の発見や整理こそが大事な場合、提案は自明なため短くなる

# イントロの3つの役割

## (と、そこから見た3点プロットとの関係)

1. 課題や提案が扱う話題への導入を行うこと
    - ◇ イントロの冒頭の背景部分でこれを行う
    - ◇ 3点プロットの背景部分を展開すれば達成できる
  2. 全体を要約して紹介すること
    - ◇ 話の全体構造を把握することで、読者や聴衆の以降の理解を円滑にする
    - ◇ 3点プロットの各項目を詳細化すれば達成できる
  3. 読者や聴衆の興味をひくこと
    - ◇ 読者や聴衆が興味を持つような点を強調する必要がある
    - ◇ 問題や提案の核心部分、華々しい結果を示すなど
      - 「200%性能向上しました」 → 「すごいな、どうやったんだろ？」
    - ◇ 3点プロットにはあまり現れないが、イントロでは意識する
- これらの役割を意識してイントロプロットを作ると良い

# 確認する内容の違い

- イントロプロットでは説明の流れの確認も行う
  - ◇ 3点プロットでは論理関係の整理に重きをおいていた
  - ◇ イントロプロットでは説明の順序/接続も確認する
- イントロプロットも1画面に収まるように書く
  - ◇ 複数画面にわたると、接続関係の確認が難しくなる
    - 3点プロットと同様の密度で書くと1画面に収まらない
    - 内容を絞る必要がある
  - ◇ トップレベルの項目同士が接続されていることを特に確認する

# 項目の接続

## ■ 項目間の接続の方法

- ◇ 各項目に、そこまでに出てきた単語/概念を必ず含める
- ◇ 過去に出てきた情報に新しい情報を付け足す形になる

## ■ 確認方法：

- ◇ そこまでに出てきた単語/概念が全く含まれていない項目が存在してしまっていないかを見る

## ■ 第1項目はそこより前がないため、これを守れない

- ◇ 第1項目は想定する読者や聴衆が既に知っているであろう概念から始める



# 例：出川くんの ICCD

1. 命令キャッシュミス減らすために多くの研究がなされてきた
    - 置換アルゴリズムやプリフェッチなど
  2. 現代では命令キャッシュミス数と実行時間が直接相関しない
    - 現代のプロセッサではミスの処理を含む様々な処理がオーバーラップされるため
    - ミス数を減らしても実行時間が短くならない場合もある
  3. 精度よい実行時間見積もりのためにはプロセッサ全体のシミュレーションが必要
  4. しかしそのような全体のシミュレーションは非常に低速
  5. 命令キャッシュミス数に代わる新たな指針を提案
    - 手法：新たな指針と、その指針を使った高速な性能見積もり
  6. 評価の結果
    - 効果：2桁短い時間でシミュレーションとほぼ同じ精度の性能見積もりを実現
- 
- ◇ 前の項目に出てきた概念/単語を次の項目の前の方で出すと上手く繋がる
  - ◇ 同じ単語/概念を結んだ矢印が基本的には左下をむく
    - 右下を向く場合、文を最後まで読まないと関係がわからない

# イントロプロットのまとめ

- 6点程度の項目からなるプロットを作る
  - ◇ 各パラグラフで何を話すかをまとめる
  - ◇ 6点の項目だけを繋げて読んでも意味が通るようにする
- イントロプロットは基本的には3点プロットから派生させて作る
  - ◇ 単に詳細化すれば良い訳ではないので注意
  - ◇ 3点プロットとの違い
    - 配分の自由度
    - 役割の意識
    - 接続の確認

# 全体プロット

---

# 全体プロット

- 論文やスライド全体のプロット
  - ◇ 特に項目数などの形式はない：
    - 基本的には3点プロットやイントロプロットから派生させて考える
    - イントロプロットでは省略されるような実装の詳細なども入る
- 3点プロットは、いわば「プロットのプロット」
  - ◇ 基本的には3点プロットで整理した内容をもとに、肉付けして全体プロットを作る
  - ◇ いきなり全体プロットを作るのは難しい

# 全体プロットはイントロプロットの単純な詳細版ではない 1

- 「イントロプロットを単純に詳細化したもの != 全体プロット」
  - ◇ 全体プロットは「基本的には」イントロプロットをより詳細化して作る
  - ◇ しかし、イントロ特有の役割により、イントロプロットと全体プロットは構造が異なる部分がある
- 「1. の話題の導入」に由来する違い
  - ◇ イントロ内で課題や提案が扱う話題への導入が済んだ後は、それ以上詳細に話す必要がない場合がある
- 「3. 興味をひく」に由来する違い
  - ◇ イントロでは問題の深刻さや提案のすごさをより強調して重きを置く
  - ◇ これは全体プロットにはあまり現れない

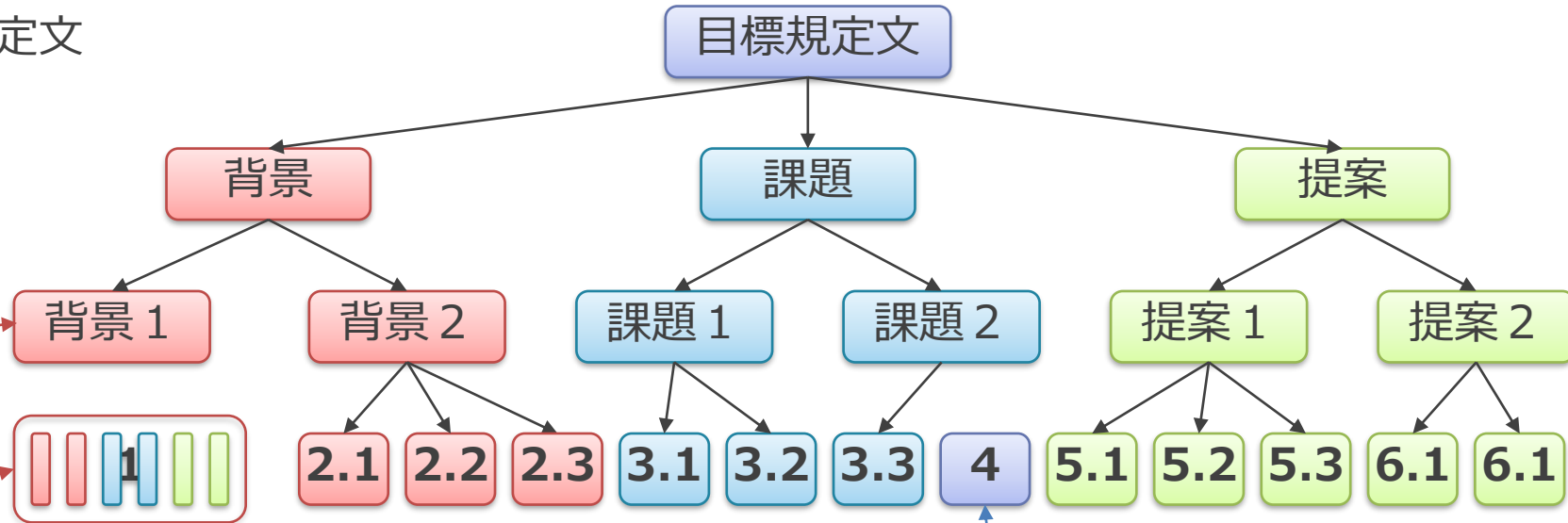
# 全体プロットはイントロプロットの単純な詳細版ではない 2

目標規定文

3点

イントロ  
プロット

全体  
プロット



- 全体プロット（章立て）は1つ上のイントロの論理構造とは1 : 1に対応しないことも多い：
  - ◇ イントロ（1章）は論文全体の概要になる
  - ◇ 背景の話題は、それ以降は話されない事も多い
  - ◇ イントロでは取り上げなかったが説明の必要がある話題が入ることもある
- 3点プロットは、大きな話題を並べる際の順序の目安ぐらいに考える

# 全体プロットの構成

- 論文用：以下を箇条書きにまとめる
  - ◇ 論文の subsection までの節タイトル
  - ◇ そこで何のために何を話すかも簡単にまとめる
    - 節同士の依存関係を明らかにする
    - 「～節で述べる～の説明の前提知識として、ここでは～を説明」など
- スライド用：以下を箇条書きにまとめる
  - ◇ スライドの各ページのタイトル
  - ◇ そこで何のために何を話すかも簡単にまとめる

# スライド用プロットの流れの例

1. イントロ
2. 背景となる問題
3. 既存手法
  1. 既存手法の説明
  2. 既存手法の問題
4. 提案手法
  1. アイデア
  2. 実装：構成，動作，例
  3. 既存手法との比較
5. 評価
6. まとめ

◇ たとえば上記それぞれの項目に 1 ～ 4 ページ程度を割り当てる



# プロットから文章へ

---

# プロットから文章やスライドへ

## ■ プロットと文章の違い：

◇ プロットは論理の階層構造を単に表せば良い

□ 子は親にぶら下がっている事で視覚的に論理関係がわかる

□ つなぎの言葉は通常あまり書かない

◇ しかし、文章（スライド）は基本的にシーケンシャル

□ 文章は前から後ろにむかって順に読むもの

□ 前から読んでわかる順序に論理を展開し、それぞれにつなぎを入れる必要がある

## ■ 課題：

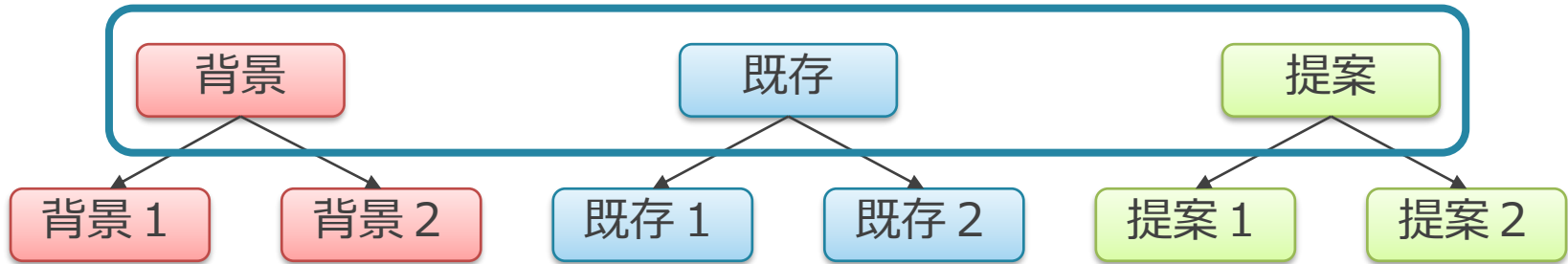
プロットの論理をどのようにシーケンシャルな文章に展開するか？

# 階層構造の展開の仕方

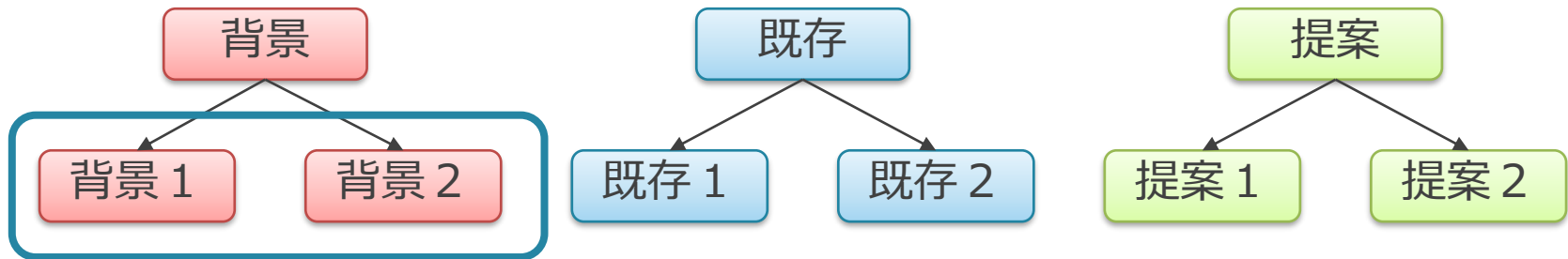
- 上から順に各階層にある話題を紹介したあと，1つずつ潜っていく
  - ◇ 典型的なやりかた：以下を再帰的に繰り返す
    - 登場人物（子）の紹介と，子同士の関係を説明
    - 各子の詳細を順に説明
- 典型例：
  - ◇ イントロで論文全体の話題を紹介
  - ◇ 2節の冒頭で背景全体を簡単に説明
  - ◇ 2.1節で背景の1つめを説明
  - ◇ 2.2節で背景の2つめを説明
  - ◇ 3節の冒頭で背景との関係と共に既存手法全体を簡単に説明
  - ◇ 3.2節で既存手法の1つめを説明
  - ◇ 3.2節で既存手法の2つめを説明
  - ◇ ...

# 上から順に各階層にある話題を紹介したあと、 1つずつ潜っていく

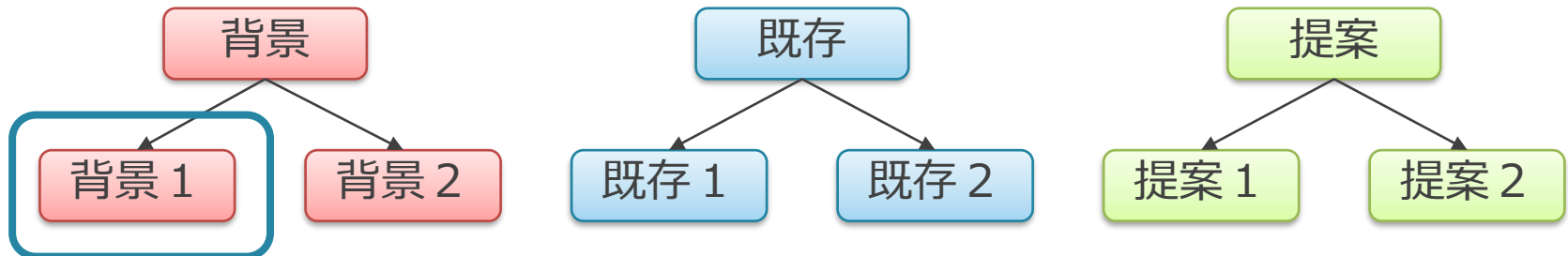
- イン트로で論文全体の話題（流れ）を紹介



- 2 節の冒頭で背景全体を簡単に説明

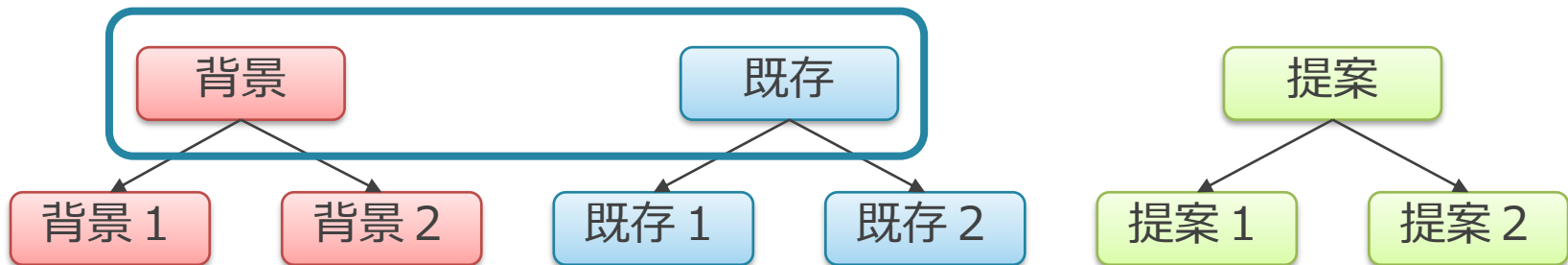


- 2.1 節で背景の 1 つめを紹介

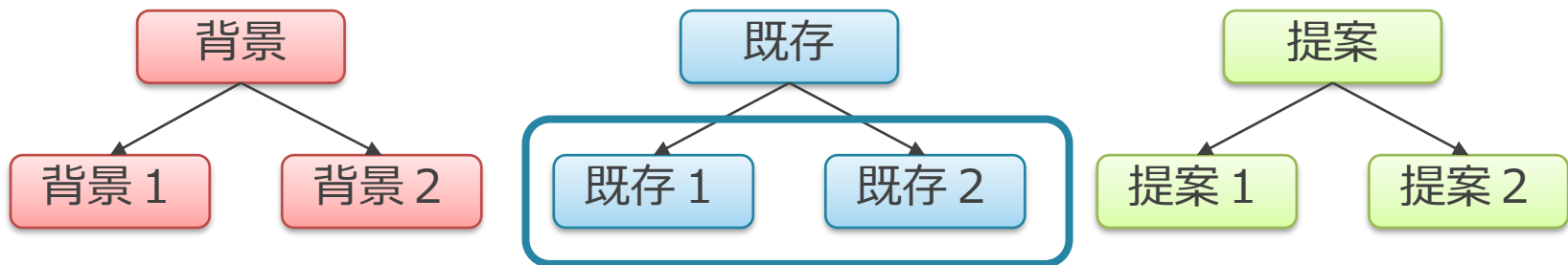


# 上から順に各階層にある話題を紹介したあと、 1つずつ潜っていく

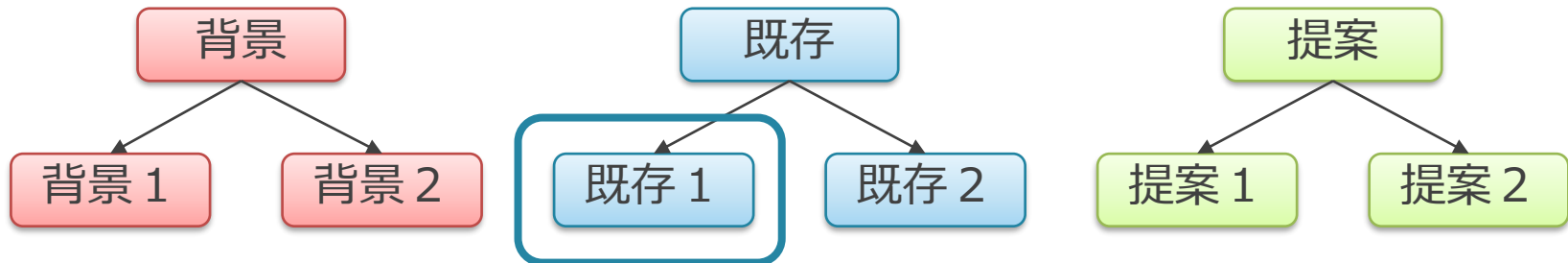
- 3 節の冒頭で背景との関係と共に



- 既存手法全体を簡単に説明



- 3.1 節で既存手法の 1 つめを説明



# イントロの書き出しについて

- アブストやイントロの先頭はトップダウンに書くことが難しい
  - ◇ 非常に一般的なことから話題を絞っていく順序で書くことになる
- ここはある種の例外だと思う

# まとめ

---

# まとめ

- プロットの作り方について説明
  - ◇ 目標規定文から全体プロットまで
  - ◇ 3点プロットから始めるとよい
    - 背景, 課題, 提案の中身と関係をはっきりさせる
- 3点プロットは基本
  - ◇ 慣れてきたらイントロプロット等から初めてもよい