ニコンの光学設計におけるAWS ParallelClusterの利用事例

インスタンス選択の考え方

株式会社ニコン
光学本部シナジー推進部基盤技術開発課
小野広起

2023年7月20日

株式会社ニコン

本日の要点

1. **ニコンの光学設計においてAWS ParallelClusterを利用した事例**
    - 光学シミュレーションにおける課題として、組み立て誤差の影響評価が挙げられる
    - オンプレミスクラスターからAWS ParallelClusterへの移行で、コスト削減や開発期間の短縮を実現
2. **AWS ParallelClusterのインスタンス選択**
    - ParallelClusterの運用ではスクリプトでクラスター構成自動化し，インスタンスタイプ変更を容易に
    - ParallelClusterでのインスタンス選択の考え方として、管理ノードと計算ノードがあり、それぞれ選択するインスタンスが異なる
    - 計算ノードではプロセスごとの処理時間のばらつきに応じてインスタンスタイプを選択
    - キューの切り替えでジョブ投入時にインスタンスタイプを変更可能
3. **今後も効率化を目指して、今後も様々なインスタンスタイプを試していく**

本日のアジェンダ

1. 発表者紹介 / 会社紹介
2. 光学シミュレーションにおける課題
3. AWS ParallelCluster インスタンス選択の具体的な取り組み
    - オンプレミスクラスターからAWS ParallelClusterへの移行
    - AWS ParallelCluser　管理ノードインスタンス
    - AWS ParallelCluser　計算ノードインスタンス
    - 人気のインスタンスタイプ
4. 光学シミュレーションにおける課題に対する効果
5. まとめ

発表者紹介

小野　広起 (ONO Hiroki)

株式会社　ニコン
光学本部シナジー推進部基盤技術開発課

光学ソフトウェアの技術サポート職を経て2002年にニコンに入社.

映像事業部のカメラレンズ設計に関わる光学シミュレーション用の環境整備,

オンプレミスとクラウドのクラスター運用，シミュレーション用ツール開発に従事.

株式会社ニコン

2030年のありたい姿
人と機械が共創する社会の中心企業

人と機械の距離を縮め、創造をもっと自由に。
共創から生まれる新しい価値を、広く社会に届けていきます。

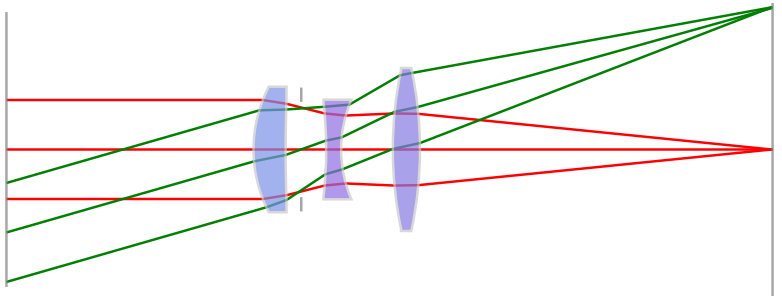
ニコンは、1917年の創業以来、一貫して高品質の製品を提供することで、世界中のお客様から信頼されてきました。
私たちの映像事業の製品は、プロの写真家や映像作家、科学者、医療関係者など、幅広い分野で利用されています。
私たちは、今後も世界中のお客様に、最高品質の製品を提供し続けるために、日々努力を重ねています。

光学シミュレーションにおける課題

- なるべく試作の前のシミュレーション段階で，開発中の製品を正しく評価したい.
- 一例を挙げると，組み立て誤差のシミュレーション
  - **目的**
    - 組み立て誤差によるレンズの結像性能への影響評価
    - 製造工程での要求精度の確認や，組み立て時の調整の容易さの確認
  - **手法**
    - 乱数を使ってカメラレンズの組み立て誤差を設定
    - 調整機構の調整量を最適化
    - 多数のケースを計算して統計処理

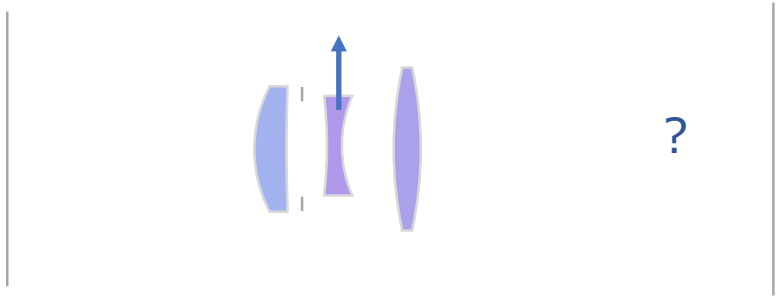
光学シミュレーションにおける課題

- **組み立て誤差のシミュレーション**
  - 組み立てに誤差があれば，設計通りの性能にはならない



  - **組み立て誤差とは**
    - レンズが傾いたり
    - レンズの位置がずれたり

  - **組み立て誤差への対策**
    - 誤差のある状態をシミュレーションで評価
    - レンズの傾きや位置を調整して性能を回復



詳細は以下の論文を参照

佐々木 豊春, 新海 雅彦, 東山 孝一郎, 田中 文基, 岸浪 建史, 鏡筒光学製品における統計的公差設計システムの開発, 精密工学会誌, 1998, 64 巻, 7 号, p. 1090-1095
佐々木 豊春, 新海 雅彦, 東山 孝一郎, 田中 文基, 岸浪 建史, 鏡筒光学製品における統計的公差設計システムの開発(第2報), 精密工学会誌, 1999, 65 巻, 2 号, p. 279-284
佐々木 豊春, 黒田 陽子, 田中 文基, 岸浪 建史, 鏡筒光学製品における統計的公差設計システムの開発(第3報), 精密工学会誌, 2000, 66 巻, 4 号, p. 572-577

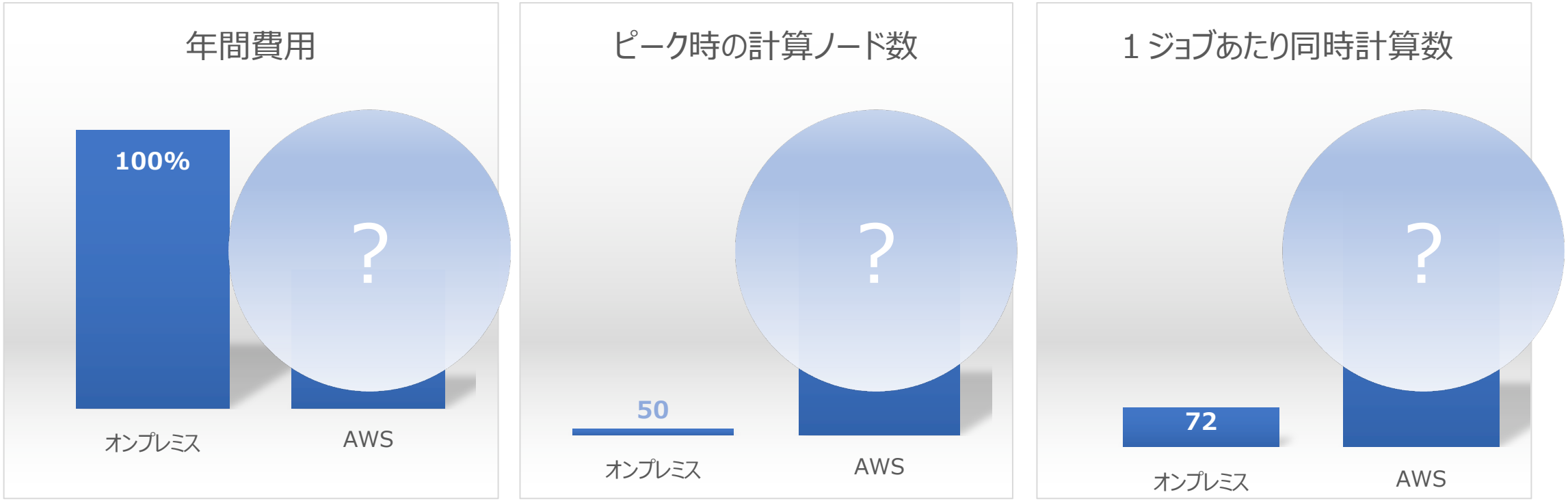
光学シミュレーションにおける課題

- 計算時間のかかるシミュレーションでは，オンプレミスの6コアCPU 50 台構成のクラスターでは，混雑して思うように複数のジョブが流れない.

- **シミュレーションのための時間を短くして，開発期間を短縮し開発費の削減をしたい**

光学シミュレーションにおける課題

オンプレミスとの比較で年間費用や計算速度についての比較を最後に紹介



本日のアジェンダ

1. 発表者紹介 / 会社紹介
2. 光学シミュレーションにおける課題
3. **AWS ParallelCluster インスタンス選択の具体的な取り組み**
    - オンプレミスクラスターからAWS ParallelClusterへの移行
    - AWS ParallelCluser　管理ノードインスタンス
    - AWS ParallelCluser　計算ノードインスタンス
    - 人気のインスタンスタイプ
4. 光学シミュレーションにおける課題に対する効果
5. まとめ

AWS ParallelCluserへの移行

- **オンプレミスクラスターからの移行は簡単**
  - オンプレミスのジョブ管理システム LSF から Slurm への移行.
  - オンプレミスと AWS ParallelCluster を並行運用

- **クラスターの規模**
  - 必要な時に必要な規模のクラスターを利用できる（予算の範囲で）

- **高速化**
  - 多くの計算ノードを利用し同時実行プロセス数を増やす事で高速化

ParallelCluster の運用

- スクリプトで CLI ツールを使い，クラスター構成など運用手順を自動化
- ジョブ実行時にユーザのデータをクラスターにコピーして実行
- ジョブ終了時にユーザのデータを回収してクラスタ上のユーザのデータを削除
- 毎月新しく複数の設定の複数のクラスターを構成し，古いクラスターを削除
- 構成したクラスターは設定変更せず，古いものを破棄して新規作成

- **結果として**
  - **ユーザのデータバックアップやデータ移行は無し**
  - **新しいインスタンスタイプを試すのが簡単**
  - **ParallelCluster の新バージョンに追従するのも簡単**

ParallelCluster 管理ノードインスタンス

- **常時起動で待機**
- シミュレーションそのものは実行しないので，そこまで処理能力が高くなくても良い
- 無料利用枠や無料トライアルの対象のインスタンスタイプを積極利用
  - t2.micro (x86_64)
  - t4g.small (arm64)
- クラスターの規模が大きくなったり，ファイル共有への要求が高くなると性能不足の場合は見直す

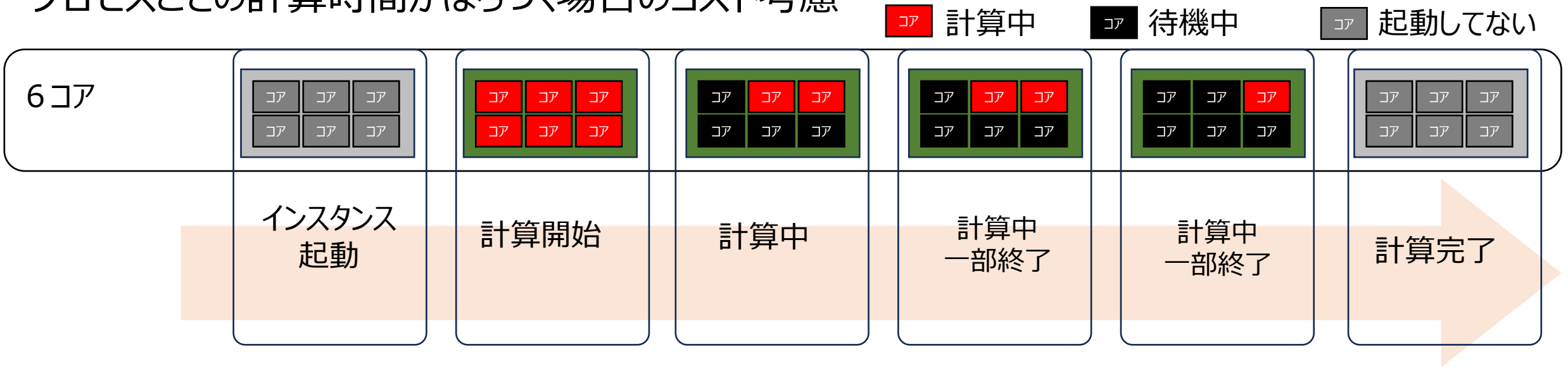
ParallelCluster 計算ノードインスタンス

- **原則としてバッチ処理で利用**
- ジョブが投入されたら起動して，計算終了で自動終了
- ParallelClusterのキューごとに異なるインスタンスタイプを設定
- ユーザがジョブの投入時にインスタンスタイプをキューで指定

- １プロセスに１物理コアを割り当て
  - x86_64 のハイパースレッディングはオフ
  - プロセスの実行時間がばらつく場合はコストを考慮

ParallelCluster 計算ノードインスタンス

プロセスごとの計算時間がばらつく場合のコスト考慮

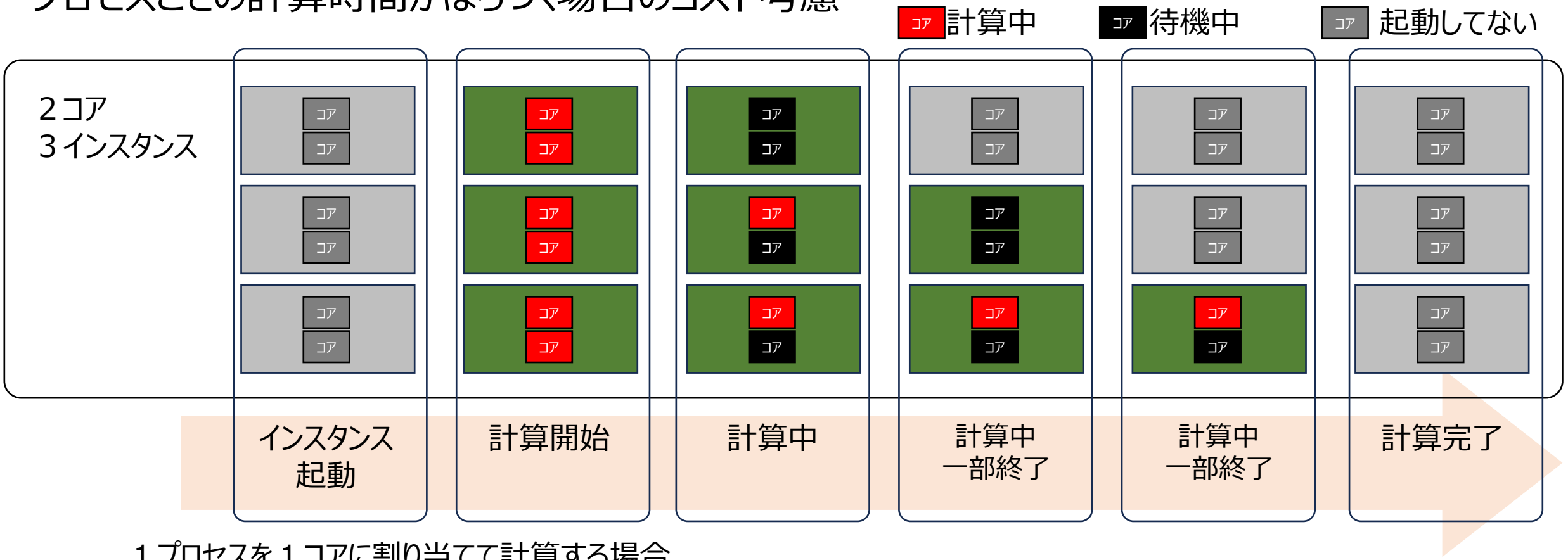


赤の計算中コアと黒の待機中のコアには課金が発生．グレーの起動していないインスタンスは課金は発生しない．

１プロセスを１コアに割り当てて計算する場合，
６コアのインスタンスでは最後の１プロセスが終わるまで，課金は変わらない．
先に計算が終わって待機中になっているコアが勿体無い．

ParallelCluster 計算ノードインスタンス

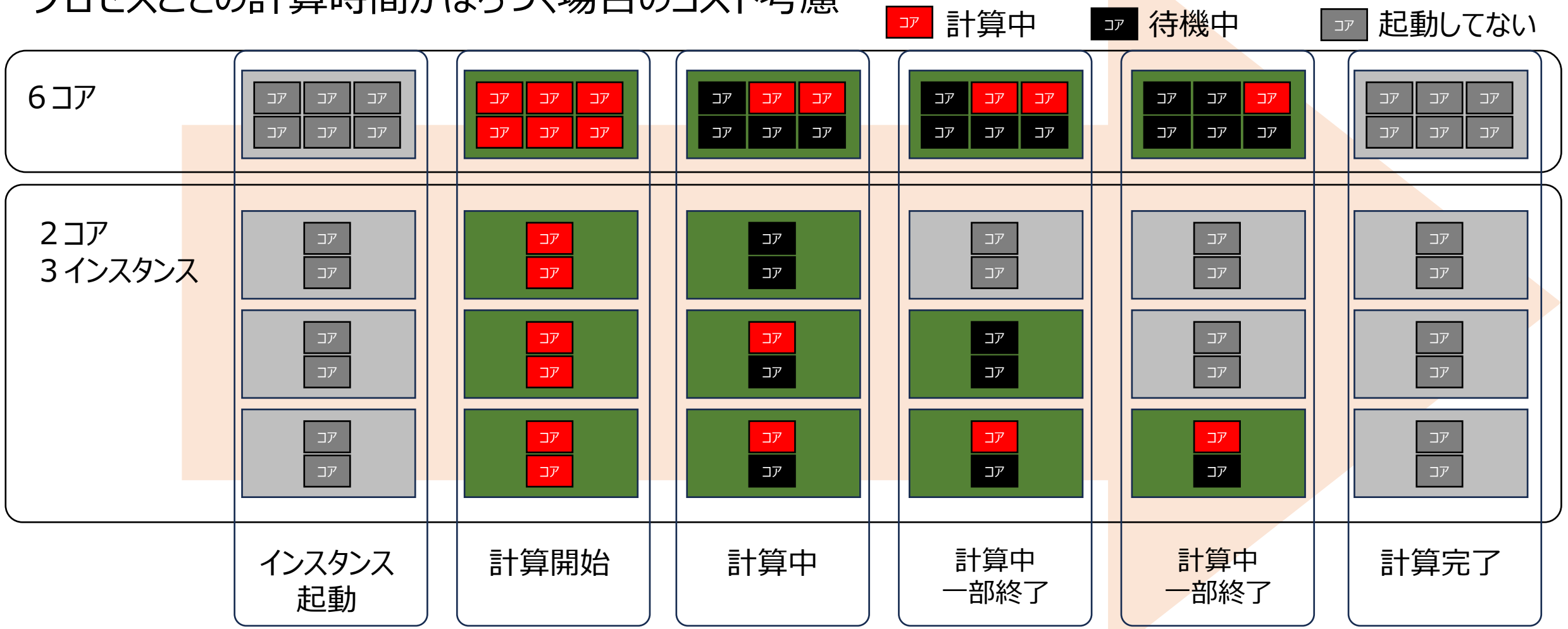
プロセスごとの計算時間がばらつく場合のコスト考慮



１プロセスを１コアに割り当てて計算する場合，
2コアの3インスタンスの構成であれば，先に計算が終わったインスタンスから終了できる.
コア数の少ないインスタンスの方が単価が安い.

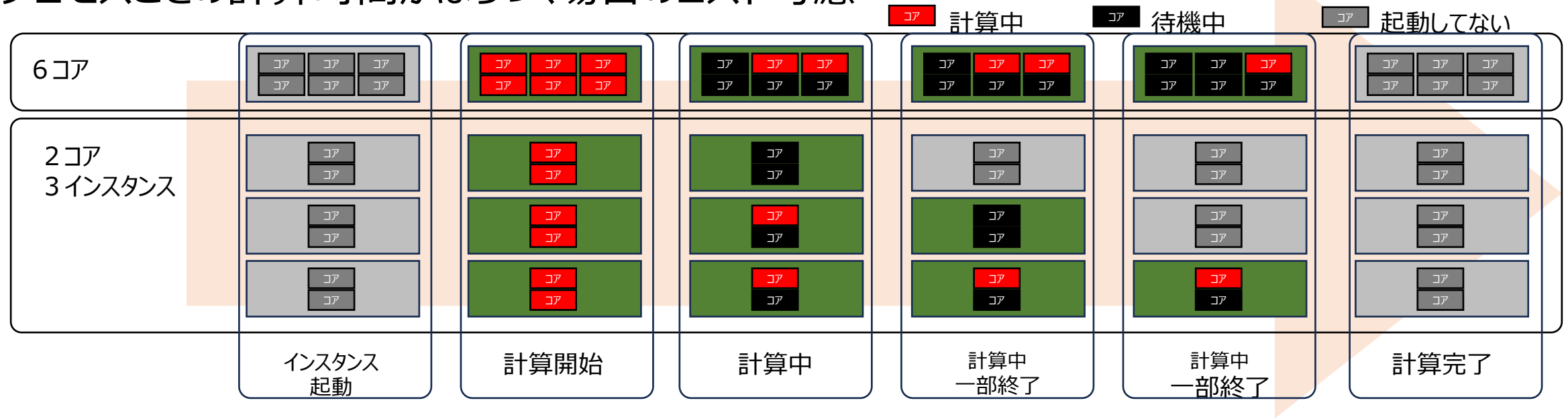
ParallelCluster 計算ノードインスタンス

プロセスごとの計算時間がばらつく場合のコスト考慮



ParallelCluster 計算ノードインスタンス

プロセスごとの計算時間がばらつく場合のコスト考慮



- プロセスごとの計算時間がばらつく場合
  コア数が多い計算ノードは，コアによって処理が長かったり短かったりになる
  計算が終わったコアから終了した方がコスト的に有利と思われるので，
  コア数の少ないインスタンスタイプを選ぶ

ParallelCluster でのインスタンスタイプ

- クラスター構成用のカスタムAMI作成時には，コア数の多いインスタンスでシミュレータをソースから並列ビルド
- 管理ノードには「一般用途向け」 t2.micro, t4g.small など
- 計算ノード
  - Intel MKL に依存しているシミュレーターなどでは z1d や c6i など
  - 分散処理するプロセスの処理時間がばらつくときは，1 インスタンスあたりの物理コアが少ない方が無駄が少ないと仮定
  - 一般的には「コンピューティング最適化」の c7g や c6i を利用
  - メモリが必要な場合は「メモリ最適化」のr7g や r6i など
  - 数十分から数時間の処理が多く，同時計算のプロセス数に対して，多くのインスタンスとCPU コアを利用できているので，HPC 向けの単価の高くコア数が多いインスタンスはそれほど必要としていない

最近のインスタンスタイプの人気

- ユーザはインスタンスタイプの違いをそれほど意識せず，色々試している
- クロックの高い z1d や m5zn を良く利用
- 最近はGraviton3 c7g は計算が速いと人気がある
- 実際に計算が速いかどうかは，

  計算ノードのインスタンスタイプの違いだけでなく
  管理ノードのインスタンスタイプの違いによる全体的な性能差などの要因がある.

本日のアジェンダ

1. 発表者紹介 / 会社紹介
2. 光学シミュレーションにおける課題
3. AWS ParallelClusterのインスタンス選択具体的な取り組み
    - オンプレミスクラスターからAWS ParallelClusterへの移行
    - AWS ParallelCluser　管理ノードインスタンス
    - AWS ParallelCluser　計算ノードインスタンス
    - 人気のインスタンスタイプ
4. **光学シミュレーションにおける課題に対する効果**
5. まとめ

光学シミュレーションにおける課題に対する効果

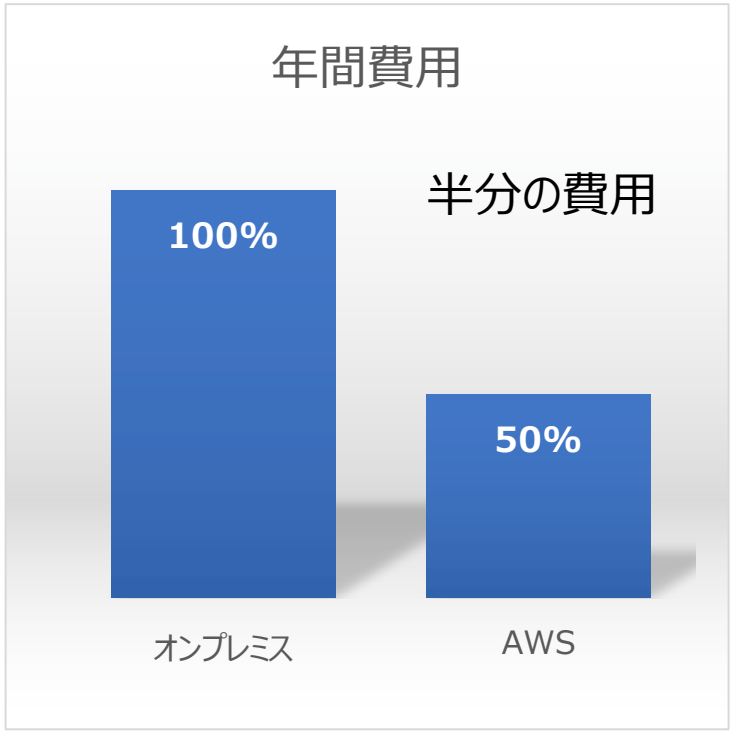
- 計算時間のかかるシミュレーションでは，オンプレミスの6コアCPU 50 台構成のクラスターでは，混雑して思うように複数のジョブが流れない.
  - **AWS でピーク時でも十分な計算ノード数を確保**

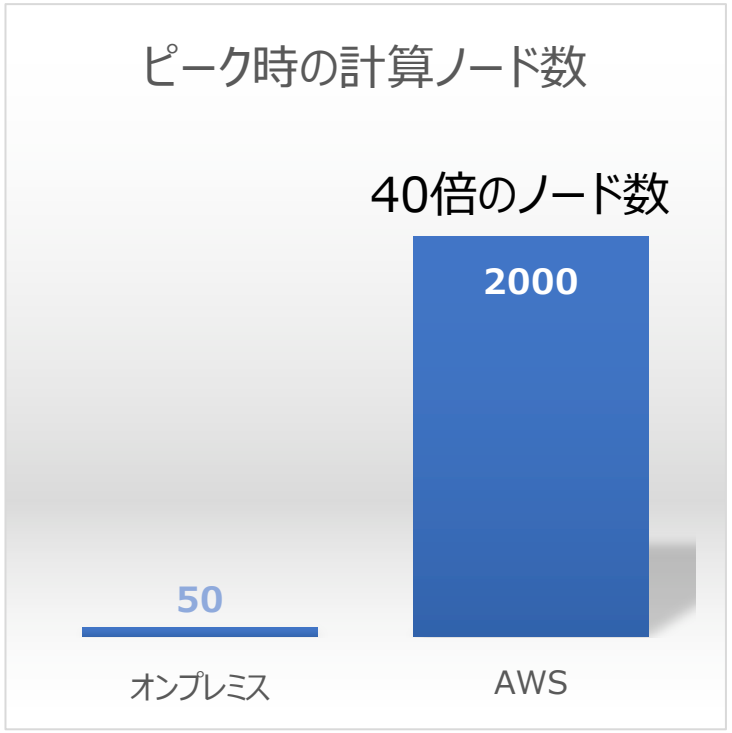
- シミュレーションのための時間を短くして，開発期間を短縮し開発費の削減をしたい
  - **年間の予算は半分に**
  - **１ジョブあたりの同時計算プロセス数が7倍で高速化**

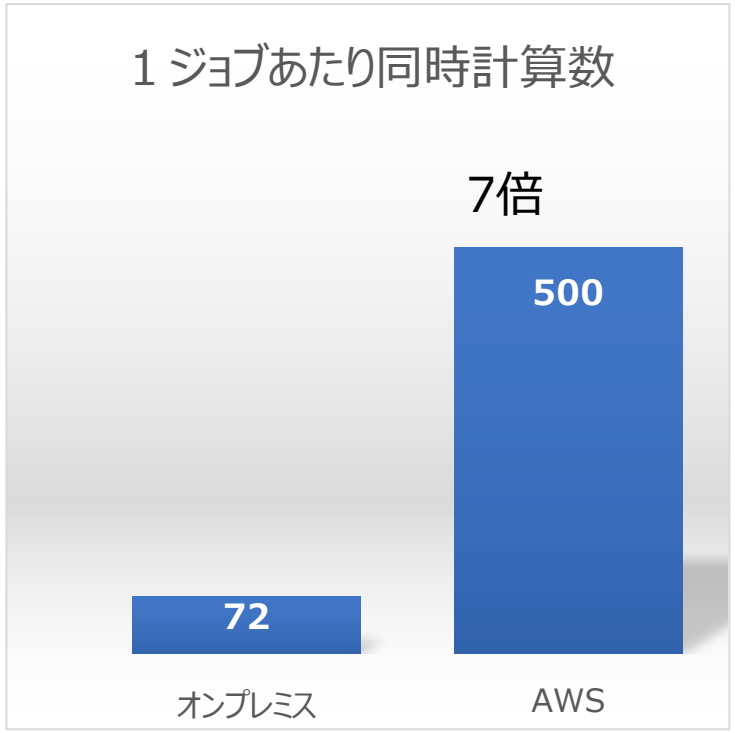
光学シミュレーションにおける課題に対する効果

オンプレミスとの比較で年間費用が半分でも，

40倍の計算ノードを利用し1ジョブあたり7倍高速に







まとめ

- オンプレミスの半分の予算で,40倍のノード数を駆使し，１ジョブあたり従来の7倍速
- 管理ノードと計算ノードでそれぞれ適切なインスタンスタイプを
- ParallelClusterでは，キューの切り替えで，インスタンスタイプを変えられる
- １プロセス１コア割り当ての並列分散処理で，プロセス数と同じ程度の数のコア数が用意できる条件では，プロセスの計算時間がばらつく場合はコア数の少ない計算ノードがコスト面では有利
- これからも様々なインスタンスタイプを試して，より効率化を目指す

今後の課題

- コストを抑える手法の探求
- シミュレーションの内容ごとに，管理ノード，計算ノードの
  それぞれの最適なインスタンスを調査してクラスターを更新していく

- **そのために必要なこと：**
  1. **管理ノードの負荷の程度を見極める**
     - 計算ノードが多くなると負荷が増えて，管理ノードの性能が全体のパフォーマンスに影響
  2. **シミュレーションの内容ごとに，プロセスの計算時間のばらつき度合いを調査**
  3. **シミュレーション実行時の，実行時間およびコストの評価とその蓄積**

ご清聴ありがとうございました.

株式会社ニコン

光学本部シナジー推進部基盤技術開発課

小野広起 Hiroki.ONO@Nikon.com


Nikon