

HyGTA2専用シミュレータ用の ログ解析ツールの使い方

コンピュータアーキテクチャ研究グループ B4

萱沼 颯

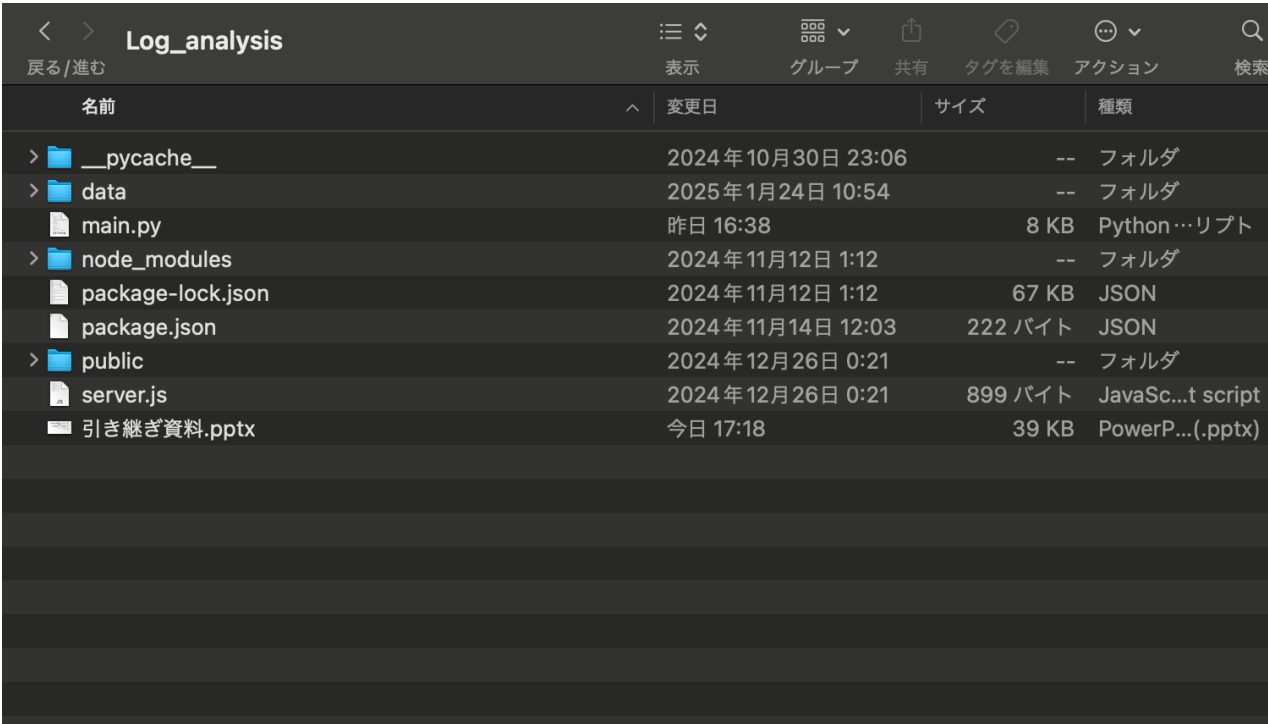
インストールが必要なもの

- Python (v3.12.2)
- Node.js (v22.9.0)

下準備（完全に初めて使う場合）

- GitHubからログ解析ツール (Log_analysis)をダウンロード
- Log_analysisのルートディレクトリにて, ターミナルから
npm installを実行

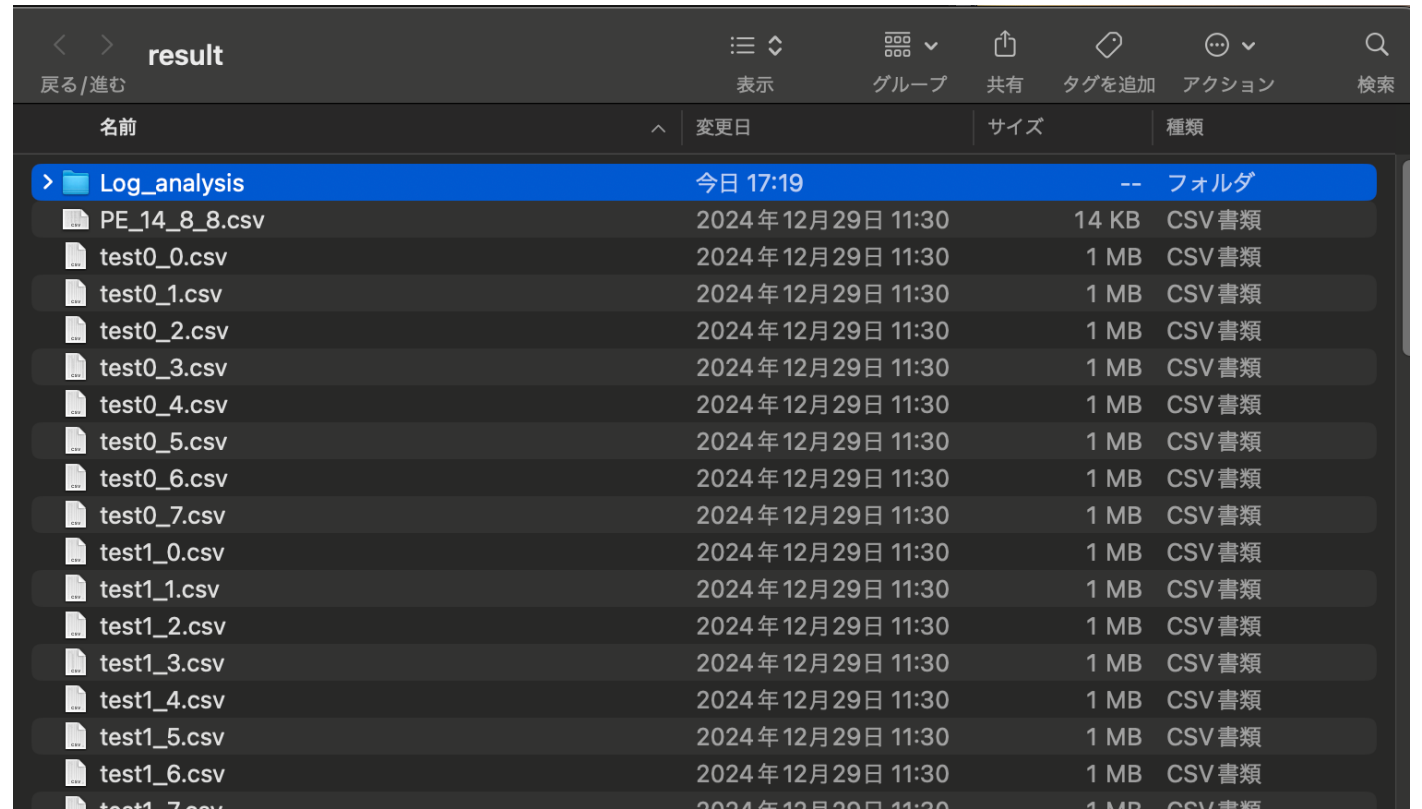
- 依存パッケージを一括
インストールする（必須）
- 実行後に”npm_modules”
フォルダが生成される



Log_analysis				
戻る / 進む				
表示 グループ 共有 タグを編集 アクション 検索				
名前	変更日	サイズ	種類	
> __pycache__	2024 年 10 月 30 日 23:06	--	フォルダ	
> data	2025 年 1 月 24 日 10:54	--	フォルダ	
main.py	昨日 16:38	8 KB	Python...リプト	
> node_modules	2024 年 11 月 12 日 1:12	--	フォルダ	
package-lock.json	2024 年 11 月 12 日 1:12	67 KB	JSON	
package.json	2024 年 11 月 14 日 12:03	222 バイト	JSON	
> public	2024 年 12 月 26 日 0:21	--	フォルダ	
server.js	2024 年 12 月 26 日 0:21	899 バイト	JavaSc...t script	
引き継ぎ資料.pptx	今日 17:18	39 KB	PowerP...(.pptx)	

使い方1-1（解析から行う場合はここから）

- ログ解析ツールをログファイル群と同ディレクトリに置く



使い方1-2

- ターミナルからpython3 main.pyを実行
 - ログファイル群の解析をするPythonスクリプト
 - 各標準入力項目については次ページ参照

```
kayanuma@makku ~ % cd /Users/kayanuma/Desktop/result/Log_analysis
kayanuma@makku Log_analysis % python3 main.py
scale:14
PE行列の行(列)数:8
モジュール数:9
アニメーションのためのファイルを作成しますか？(T/F):T
データをとるモジュールindex(1~):4
各深さのサイクル数をカンマ区切りで入力してください(例: 25,389,2668,841,34) (わからない場合はそのままEnter): 63,2803,11242,1532,32
分析中...
完了しました
kayanuma@makku Log_analysis %
```

Python3 main.py時の標準入力について1

- モジュール数 . . . PE内部のモジュール数のこと .
csvファイルでは列数のこと（現状9） .
 - データをとるモジュールindex . . . 稼働率平均値/中央値/分散
をどのモジュールでとるか（迷ったら4） .
 - 1 : local frontier bitmap
 - 2 : working frontier
 - 3 : extract vertices issue receiver
 - 4 : traversal
 - 5 : filter pred data issue
 - 6 : unvisit
 - 7 : filter pred data issue receiver
 - 8 : Allgather
 - 9 : AlltoAll
- 9列

	test0_0								
0	Send	Wait	Wait	Wait	Wait	Wait	Wait	Wait	Wait
1	Wait	Receive	Wait	Wait	Wait	Wait	Wait	Wait	Wait
2	Wait	Send Allgather	Wait	Wait	Wait	Wait	Wait	Receive	Wait
3	Wait	Wait	Wait	Wait	Wait	Wait	Wait	Send Frontier	Wait
4	Wait	Receive Alltather	Wait	Wait	Wait	Wait	Wait	Wait	Wait
5	Wait	Send -1	Wait	Wait	Wait	Wait	Wait	Wait	Wait
6	Wait	Wait	Receive	Wait	Wait	Wait	Wait	Wait	Wait
7	Wait	Wait	Wait	Send End	Wait	Wait	Wait	Wait	Recieve End

[illegible]

Python3 main.py時の標準入力について2

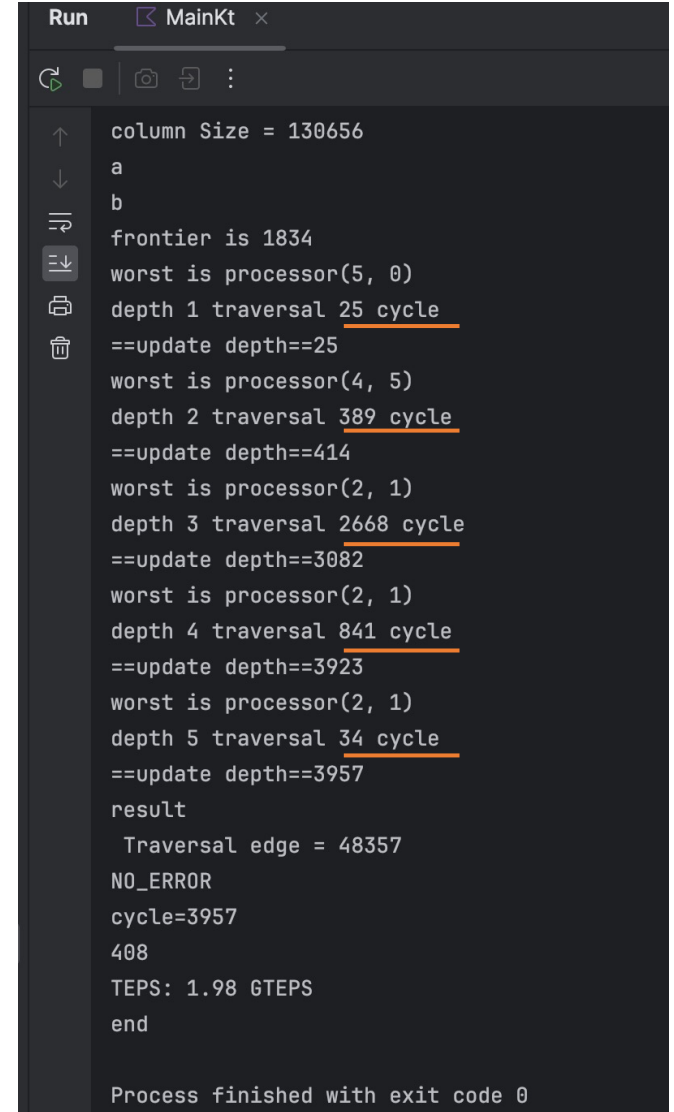
- 各深さのサイクル数・・・
専用シミュレータのターミナルから情報取得.

例えばPEアレイ8×8の64の場合
(PE数とグラフスケールによって変わる)

Scale10
18,97,706,262,23

Scale12
25,389,2668,841,34


Scale14
63,2803,11242,1532,32

A screenshot of a terminal window titled 'Run' and 'MainKt'. The terminal displays the output of a simulation. The output includes: 'column Size = 130656', 'a', 'b', 'frontier is 1834', 'worst is processor(5, 0)', 'depth 1 traversal 25 cycle' (underlined), '==update depth==25', 'worst is processor(4, 5)', 'depth 2 traversal 389 cycle' (underlined), '==update depth==414', 'worst is processor(2, 1)', 'depth 3 traversal 2668 cycle' (underlined), '==update depth==3082', 'worst is processor(2, 1)', 'depth 4 traversal 841 cycle' (underlined), '==update depth==3923', 'worst is processor(2, 1)', 'depth 5 traversal 34 cycle' (underlined), '==update depth==3957', 'result', 'Traversal edge = 48357', 'NO_ERROR', 'cycle=3957', '408', 'TEPS: 1.98 GTEPS', 'end', and 'Process finished with exit code 0'.

```
Run MainKt x
column Size = 130656
a
b
frontier is 1834
worst is processor(5, 0)
depth 1 traversal 25 cycle
==update depth==25
worst is processor(4, 5)
depth 2 traversal 389 cycle
==update depth==414
worst is processor(2, 1)
depth 3 traversal 2668 cycle
==update depth==3082
worst is processor(2, 1)
depth 4 traversal 841 cycle
==update depth==3923
worst is processor(2, 1)
depth 5 traversal 34 cycle
==update depth==3957
result
  Traversal edge = 48357
NO_ERROR
cycle=3957
408
TEPS: 1.98 GTEPS
end
Process finished with exit code 0
```

使い方1-3

- うまく実行できるとdataフォルダに4ファイル（3ファイル）生成される
 - each_module_oc_rate.json・・・各PE内部の各モジュールの稼働率を格納
 - frames.json・・・PE挙動アニメーション用（作成しないモードでは生成されない）
 - info.json・・・色々な情報を格納. 特に指定したモジュールの稼働率平均値/中央値/分散が average/median/varianceに記載.
 - traversal_layers.json・・・各PEの指定したモジュールに対して, 各深さでの稼働率を格納

< > data		⋮ ↕
戻る/進む		表示
名前	^	変更日
 each_module_oc_rate.json		今日 17:46
 frames.json		今日 17:46
 info.json		今日 17:46
 traversal_layers.json		今日 17:46

使い方2-1（閲覧のみ行う場合はここから）

- ターミナルからnpm startを実行
 - ローカルホストでサーバを設置し、ブラウザからアクセスしてくれる。

