eBPF を用いた低遅延電力・温度モニタリング基盤の設計と実装

2022TC065 對馬秀悟
指導教員：宮澤元

1　はじめに

本研究では, カーネル空間での温度, 電力抑制スケジューリングを実装し, 体感を損なわない応答時間で運用可能であることを示す.

近年の OS は温度や負荷に応じた周波数制御や冷却制御を備えるが，制御の粒度や決定遅延により，応答性と電力の両立が難しい場面が残る．本研究では，カーネル内に観測点を設けて温度・負荷等を即時に取得し，共有データ構造に保持する．スケジューラはこの情報を基に高温・高負荷時の処理配分を滑らかに切り替える．ユーザー空間を経由しないことにより，意思決定から適用までの遅延とオーバーヘッドを最小化することを狙う．本研究の貢献は以下の点である．

- カーネル内で完結する高温防止制御を提示し, 試作を行う
- 試作したシステムを用いて温度, 消費電力, 応答時間とスループットを評価する

2　背景

近年, エッジ環境や小規模クラスタでは, 限られた冷却・電力予算の下で応答性とスループットを両立する運用が求められている.

エッジ環境や小規模クラスタでは，電力・冷却・空間の制約が厳しく，遅延要求も厳格であるため，エネルギー効率とサービス応答性の同時最適化が求められる．Linux には温度トリップポイントに応じて周波数制御や冷却デバイスを動かす Thermal　Framework ／ CPUfreq といった仕組みが用意されているが温度や使用率による判別であり, タスクの状況に応じた処理には対応しきれていない.

また, 近年カーネル空間に小型のプログラムを差し込むことで機能を拡張できる機能である eBPF が注目されている.Kepler は eBPF とセンサ情報等を活用して，コンテナ／ Pod 単位でエネルギー関連メトリクスを推定・輸出するプロジェクトである [2] が, 制御までは行っていない.eBPF を使用した監視ツールは多く実装されており, 制御面でも昨年 eBPF を用いた拡張スケジューラである SCX が実装された.

本研究はこれらの流れを踏まえ，監視から配分決定までをカーネル内に閉じる設計により制御遅延とオーバーヘッドを抑えつつ，タスク特性に応じた段階的縮退で高温・高負荷時の安定動作を実現することを狙う.

3　設計

本研究ではメトリクスを取得するプログラムとスケジューリングを変更するプログラムの２つを作成する.

3.1　システム

本研究のシステムを図１に示す.

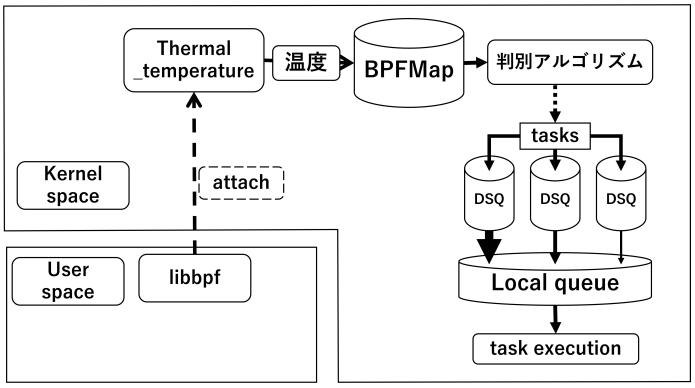

図１　システムの構成

図に登場している DSQ とは SCX の機能の１つであり, タスクをためておくことができるキューである.

メトリクス取得側　tracepoint/thermal/thermal_temperature をフックし，温度段階を決定して BPF マップに保存する.

スケジューリング側　SCX で BPF マップから温度を参照し，必要に応じ負荷軽減を実施する.

3.2　仮説

以下に温度, 電力抑制が見込める理由を記す.

タスクはローカル DSQ に入っており、そこから取り出されることで実行される.SCX ではローカル DSQ に入る前のタスクの置き場であるカスタム DSQ を作ることができる. これは任意個作ることができ、どこにタスクを入れるのかは自由に決めることができる．本研究では最終的に特定ラベル専用の常時参照キューとスコア別のキューを設計する. 温度と消費電力量に応じてローカル DSQ に吐き出す量を切り替える.

従来の温度, 電力抑制の手段は周波数や予測待機時間などによる判別が主流でタスクの特性は考慮していないが, 提案手法ではそれらを考慮したうえで先に処理すべきものを判別できる.

4　実装

以下表１に実験環境を示す.

表 1 実験環境

使用機器	Raspberry Pi 5 Model(メモリ 8GB)
バージョン	Linux raspberrypi 6.12.34-v8-16k
SCX バージョン	v1.014-495-g8d646274

4.1 現状

現状の負荷軽減方法は BPFMap を参照し, 予め決めた閾値を跨いだら

- 参照する FIFO キュー本数の縮退
- タイムスライスの短縮
- 上位キューの優先的な参照

などを行う.
スコアリングで上位キューに入れるのか下位キューに入れるのかを決める. 本来は複数軸で行うが今回は CPU の待ち時間のみを対象にしている.

ただし, 現状のプログラムはメトリクス取得と SCX の動作や連携の確認のためのものである.

4.2 今後の展望

今後の改善点としては複数軸による判断, タスクをキューに割り振る際のスコアリングの負荷, 負荷対策がほぼ一意的に決まってしまう点, 高温時は下位キューを全く参照しないなどがある. 特にスコアリングはタスク毎に計算する必要があるので高負荷時にはかなり影響してしまう可能性がある. 対処するためには

- ラベルの導入
- 優先処理をするキューの導入
- 高負荷時の負荷計算方法の見直し

が有効であると考えられる. 上 2 つはそれ単体での実装は済んでいる. そのため後は現状の方法との相性の確認と負荷の計算方法を詰めていく必要がある.

5 実験

温度ピーク抑制量とスループット低下のトレードオフ, および遅延のばらつきを評価観点とする. 以下表 2 に I/O 系の負荷をかけた実行結果を示す.

表 2 実行結果

	作成したプログラム	デフォルト
温度	60.05 ℃	59.5 ℃
読み込み速度	2198KiB/s	2528KiB/s
書き込み速度	951KiB/s	1094KiB/s
実行時間	130.064s	113.1s

今回は fio コマンドを用いて負荷をかけた. 現状のプログラムは期待通りの挙動をしない. 具体的には I/O 系の負荷をかけたときに実行時間も温度も通常時より高くなっている. これはタスク割り当て時の基準で I/O が後回しにされがちとなるためである.

表 2 からも分かるようにデフォルト時のほうがスループットは約 15% 高く, 実行時間は約 13% 短くなっている. また,I/O 以外の負荷の場合は温度は約 1 ℃下がり, 実行時間は数 % 伸びた.

今後はこれらの問題点に対処しつつ温度低下とスループット向上に努めていく.

6 おわりに

本稿では eBPF と SCX によりカーネル空間で温度指標に応じてスケジューリング挙動を制御する試作系の設計と実装要点を示した.

今後は電力測定機器の追加を行い, プログラムの基本動作の検証を終えたため, 今後は設計を再構成し, 安全性・保守性・拡張性を備えた本実装へ移行する.

参考文献

[1] eBPF”eBPF Documentation” https://ebpf.io/what-is-ebpf/ (accessed September 29.)

[2] Amara,M, et.al: ”Process-based Efficient Power Level Exporter” *2024 IEEE 7th International Conference on Cloud Computing (CLOUD).*

[3] The Linux Kernel ”Extensible Scheduler Class” https://docs.kernel.org/scheduler/sched-ext.html (accessed September 29.)