RUST プログラムのカーネル空間における配列アクセス性能に関する一考察

大河俊介 [1]　山口実靖 [1]

1. はじめに

近年，OS(Operating System，オペレーティングシステム)のカーネルはアセンブリ言語の様な低級言語や，C 言語の様な高級言語でない言語で実装されることが多かった．主要なカーネルの一つである Linux でも，2022 年までこの指針に基づき実装されてきた．一方で，2022 年 12 月に公開された Linux 6.1 以降では，開発言語に Rust[1]言語が追加され，Rust 言語によるカーネル機能(モジュールなど)の開発が可能になるとともに，一部の機能が Rust 言語で実装され公開されるようになった．

Rust 言語は，コンパイル時や実行時におけるバッファオーバーランの検出機能を有しており，C 言語など比較しバッファオーバーランなどを引き起こすプログラミングのミスに対して安全である．一方で，実行時にバッファオーバーランの検出を行う場合は，そのオーバヘッドにより性能の低下が生じる可能性があると予想される．本ポスター発表では，Rust の性能を評価し，考察を行う．特に，カーネル空間で動作するプログラムの性能についての考察を行う．

2. Rust

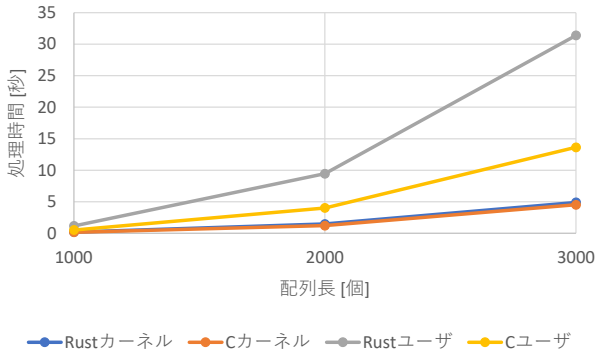
Rust[1]は Mozilla 社によって 2010 年 7 月に開発されたプログラミング言語であり，C 言語などの言語と比較し，メモリアクセスの安全性を高める仕組みが施されている．安全性の高さなどの理由により，Linux カーネルなどのカーネルプログラミングにも採用されている．

3. 性能評価

以下の配列アクセスプログラムを Rust 言語および C 言語で実装し，それをユーザ空間およびカーネル空間で動作させ，性能を調査した．

配列アクセスプログラム: 長さ 1000 から 3000 の int 型配列を用意する．また，引数に配列と整数を受け取り，その整数をインデックスとする配列の要素を戻り値として返す関数を用意する．そして，配列の全要素に対して同関数を呼び出して戻り値を得る時間を評価する．同関数内では引数として受けとった整数がインデックスとして適切か(配列サイズの範囲内に含まれるか)をコンパイル時に静的に決定することはできない．よって，オーバーラン検出を実現するには実行時の動的な確認が必要となる．

評価の結果，カーネル空間における実行においては言語によらず実行時間がほぼ同一であることが分かった．また，ユーザ空間による実行よりもカーネル空間における実行の方が処理時間が短いことが分かった．これは，カーネル空間における動作が CPU の特権モードで行われることや，コンパイラの最適化などの差によるものと思われる．



4. おわりに

本ポスター発表では，Rust 言語の性能，特にカーネル空間における性能に着目し，考察を行った．結果，Rust 言語における配列アクセスなどにて性能低下が確認される場合があることを分かった．今後は，より複雑な動作の性能についての評価やコンパイル済みバイナリの調査，性能改善手法についての考察を行っていく予定である．

謝辞　本研究は JSPS 科研費 21K11854, 21K11874 の助成を受けたものである．

参考文献

[1] Nicholas D. Matsakis and Felix S. Klock. 2014. The rust language. Ada Lett. 34, 3 (December 2014), 103–104. https://doi.org/10.1145/2692956.2663188

1 工学院大学　Kogakuin University