

Rust & calcul scientifique

Rencontre thématique Calcul Scientifique 2024, INRIA

2024-10-10

Guilhem Saurel



Figure 1: IR en robotique humanoïde

- Performance
- Fiabilité
- Productivité

- Un compilateur bienveillant
- Un gestionnaire de paquet moderne
- Du formattage et de l'analyse statique

- Un compilateur bienveillant
- Un gestionnaire de paquet moderne
- Du formattage et de l'analyse statique

Tout intégré et par défaut: Cargo

Cargo n'est bon que pour le rust.

Dès qu'on doit se linker à autre chose, il faut un gestionnaire de paquet plus général.

Safe Rust guarantees an absence of data races, which are defined as:

- two or more threads concurrently accessing a location of memory
- one or more of them is a write
- one or more of them is unsynchronized

- plusieurs accès en lecture seule

XOR

- un seul accès en écriture

- plusieurs accès en lecture seule

XOR

- un seul accès en écriture

Bonus: `const` by default

Learning curve...

“No way to prevent this” say users of only language where this regularly happens

— <https://xeiaso.net/blog/> x 12

```
enum Result <T, E> {  
    Ok(T),  
    Err(E),  
}
```

```
enum Option <T> {  
    Some(T),  
    None,  
}
```

```

let file = File::create("data.txt");
match file {
    Err(e) => eprintln!("cant create file: {e:?}"),
    Ok(mut f) => {
        let ret = f.write_all(b"hello");
        match ret {
            Err(e) => eprintln!("cant write: {e:?}"),
            Ok(_) => println!("file written."),
        }
    }
}

```

```
fn main() -> std::io::Result<()> {  
    let mut file = File::create("data.txt");  
    file.write_all(b"hello");  
    println!("file written");  
    Ok::()  
}
```

```
fn sum_of_squares(input: &[i32]) -> i32 {  
    input.iter()  
        .map(|&i| i * i)  
        .sum()  
}
```

```
fn sum_of_squares(input: &[i32]) -> i32 {  
    input.iter()  
        .map(|&i| i * i)  
        .sum()  
}
```

```
fn sum_of_squares(input: &[i32]) -> i32 {  
    input.par_iter()  
        .map(|&i| i * i)  
        .sum()  
}
```

Tooling d'autres langages

- python (ruff, uv)
- javascript (deno, parcel, oxc)

Système

- linux
- coreutils
- shell utils

Avoid dynamic libraries for now.

- <https://doc.rust-lang.org/book/>
- <https://doc.rust-lang.org/nomicon/>

- <https://doc.rust-lang.org/book/>
- <https://doc.rust-lang.org/nomicon/>
- <https://wiki.2rm.cnrs.fr/AnfRust2024>

Inscriptions ouvertes jusqu'à 18h !

Contact

Matrix: @gsaurel:laas.fr

Mail: gsaurel@laas.fr

Cette présentation

[https://homepages.laas.fr/gsaurel/talks/
rust-hpc.pdf](https://homepages.laas.fr/gsaurel/talks/rust-hpc.pdf)

[https://gitlab.laas.fr/gsaurel/talks :
rust-hpc.md](https://gitlab.laas.fr/gsaurel/talks :
rust-hpc.md)



<https://creativecommons.org/licenses/by-sa/4.0/>