Coding guideline : When dealing with large data in the output of a function should it be return value or argument with reference? #3209

yukkysaito started this conversation in **General**

yukkysaito on Jan 17, 2023 Maintainer edited

Background : autowarefoundation/autoware.universe#1947 (comment)

I would like to discuss the coding policy when outputting large data in a function.
In the current code, the following two types of functions are employed.

- Argument with reference
  - sample1
  - sample2
  - sample3
- Return value
  - sample1
  - sample2

For readability, the return value is better. But, the reason for not using the return value is that memory copying may occur.
Memory copying affects execution time and squeezes memory bandwidth. In TIER IV, in-vehicle computers could not process due to insufficient memory bandwidth.
Each of these has its own good and bad points.

Argument with reference

- Good points
  - Major codes use argument with reference.
    - OpenCV
    - ROS
    - PCL
  - No memory copy
- Bad points
  - Return value is more readable

Return value

- Good points
  - Good readability
- Bad points

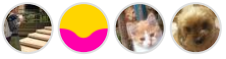
Category

General

Labels

None yet

4 participants



- Memory copying may occur because NRVO is guaranteed by compiler
  - NVRO is non-mandatory

If the data size is small, readability should be a priority, but if the data size is large, we need opinions on what should be prioritized.
We need opinions not only from algorithm engineers, but also from embedded computing engineers and others.

↑ 1 👀 2

4 comments · 2 replies

Oldest | Newest | Top

kenji-miyake on Jan 19, 2023

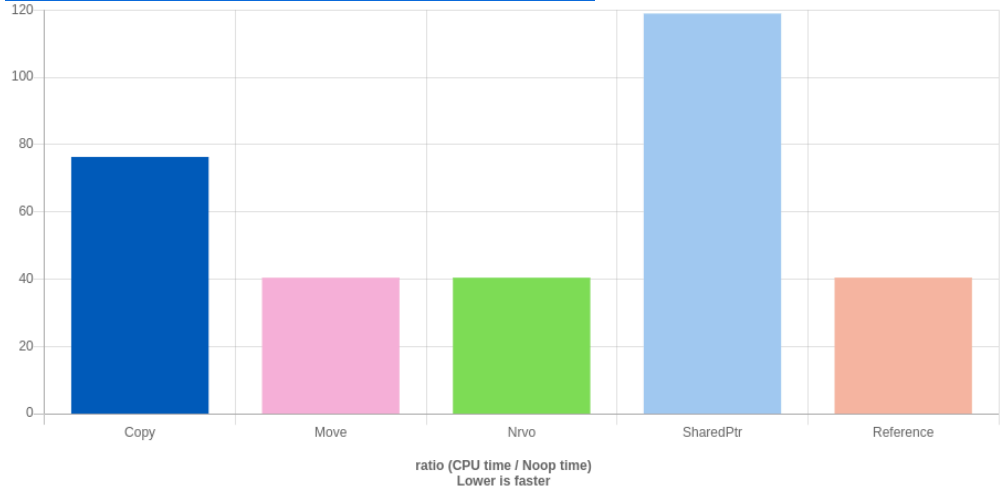
@yukkysaito Thank you for creating this discussion thread!
I believe this kind of topic is good to be discussed with measurement data.
As we investigated this before, I'll share the results.

First, there are several cases that `NRVO` isn't used:

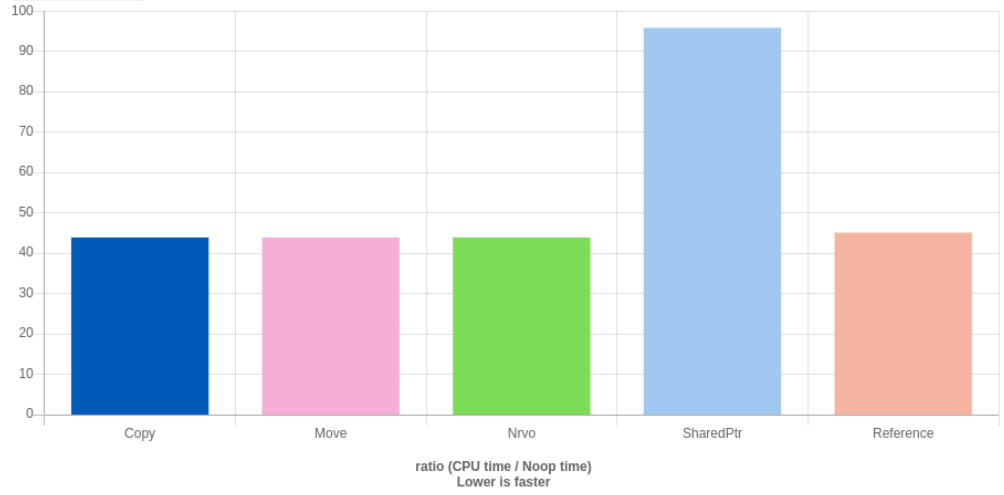
- The return value is unnecessarily wrapped by `std::move()`.
  - This can be detected by the compiler.
- There are multiple `return` statements.

However, even if `NRVO` isn't used, `move` is usually used before `copy`:
https://wandbox.org/permlink/H9yFMi7vVLy1bZQf

Also, the performance of `move` is not so bad (`shared_ptr` has a worse performance than I thought): https://quick-bench.com/q/BoCkza8cUfj2Z0esLbJAfdjQGIc



FYI, with Clang, even `copy` is optimized and `move` is a little faster than `reference` : https://quick-bench.com/q/haJwVQgTB1BHQT5gFoHodfcAoqI)



Based on these results, I recommend using `return values` as recommended in the C++ Core Guidelines.

Please let me know if there are any mistakes in my investigation methods or results.

↑ 2 👀 1 1 reply

yukkysaito on Jan 22, 2023 Maintainer Author

Thank you for comment.
I think these results will be good for the conclusion. 👍

kenji-miyake on Jan 19, 2023

> Major codes use argument with reference.

Regarding this part, I think it's not fair to only reference the projects mainly written before C++11. They may have guidelines that differ from modern standards.

↑ 1 👍 1 0 replies

TakaHoribe on Jan 19, 2023 Maintainer edited

I have doubts about the potential drawbacks of using return-value. No negative behavior was found in either the report from @kenji-miyake or my investigation. (It is known the NRVO is not guaranteed, but the compiler will optimize it anyway, and maybe it calls `move` in that case.)

Additionally, there are some popular guidelines suggesting the use of return-value.

- C++ Core Guidelines
- Google's C++ Style Guide

Unless evidence of a significant problem arises, there seems to be no benefit in deviating from these guidelines and using a different approach.

Note: of cause, there are cases where using reference value is preferred, for example, modifying a few fields in a large object. However, it can not be a reason to prohibit using return-value in all situations. It totally depends on each implementation.

↑ 1 👍 3 0 replies

yukkysaito on Jan 22, 2023 Maintainer Author edited

In conclusion, even if the data is large, I would go in the direction of using return values.
Ref :
https://github.com/orgs/autowarefoundation/discussions/3209#discussioncomment-4723928
@xmfcx @mitsudome-r Any comments?

cc @sykwer @veqcc @isamu-takagi

↑ 1 👍 3 1 reply

xmfcx on Jan 22, 2023 Maintainer

Yeah, I generally go with simple return statements and hope RVO does its job.

Another discussion point could be to talk about handling multiple return values:

- return tuple
- return struct
- return with editing passed references

But I think I'm ok with leaving this to developers to decide.