

---

Subject: Re: Impressive improvement in `std::vector` when dealing with raw memory.  
Posted by [Lance](#) on Sat, 19 Nov 2022 21:04:47 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

Morale of `basic_string` story: sometimes (more than occasionally), it's possible to make a class trivially relocatable by slightly changing your design.

While I did not do a speed comparison of the underlying memory copy facilities (just move a volume of bytes around repeatedly for certain times) of `std::vector` and `Upp::Vector`, an intuitive explanation of `Upp::Vector`'s performing well on small memory size and lagging behind when the memory block getting large is the difference in their respective memory management strategies.

A `std::vector` doubles it's capacity at each growth (until out of memory etc) while a `Upp::Vector` grows by 1/3 of its current capacity. `Upp::Vector` mitigates its supposedly more frequent allocation/relocation by doing `TryRealloc`, which, when success, housed objects relocation can be avoided. While the latter has more chances to succeed when allocated memory block is small (thus results in a amortized gain over `std::vecotr` ), it tends to fail more often when the allocated memory block is big. In which case more frequent reallocation and relocation plus additional cost on (almost bound to fail) `TryRealloc`(will have to lock some mutex at least) drag the overall performance.

---