Subject: Re: A new container in works: Flex - fast insertion vector
Posted by sergei on Fri, 21 Sep 2007 12:51:17 GMT
View Forum Message <> Reply to Message

luzr wrote on Fri, 21 September 2007 14:30sergei wrote on Fri, 21 September 2007 07:51When I said sorted containers what I needed is not find/unique (hash could do that too), but to iterate over all elements, from smallest ro largest / reverse. You say operator[] is slow. Would such iteration be slower than in set, or not?

Not actually measured, but iterating through link structures tends to be slow with modern CPUs, so I would say that the linear iteration would be about the same.

That said, it is a bit surprising that you would prefer sorted container only to get items in order. Using Index (or generally a hash_map) and Sorting the result beats this in most cases.

For me, the main advantage of such container would be the lower and upper bounds.

Quote:
I don't know how allocators work, but what have you done to get it faster than default? Some kind of buffering to allocate once for several small elements?

The algorithm of latest U++ allocator is described in the Core/srcimp.tpp.

Mirek

Sorting would beat it for a static set. But would it, if the set was changing (adding/removing elements), and at any given time I might need the order? Add/remove is O(logn), iteration should be linear, and sorting would cause the iteration to become O(nlogn) (sort+iterate), unless add/remove maintains sort order. And even in the best case of flex, add/remove would be O(sqrtn), worse than O(logn), right?