<https://github.com/nlohmann/json/issues/1924>

I had reproduce successful in ubuntu.  
My json data size about 1.4 KB but i parsed this json data for many times.  
Here are my test result:

Before Run:

KiB Mem : 8167476 total, 5461204 free, 284120 used, 2422152 buff/cache

1000 times:

KiB Mem : 8167476 total, 5456600 free, 288724 used, 2422152 buff/cache

10000 times:

KiB Mem : 8167476 total, 5405916 free, 339376 used, 2422184 buff/cache

100000 times:

KiB Mem : 8167476 total, 4893176 free, 852104 used, 2422196 buff/cache

After input the int (After run)

KiB Mem : 8167476 total, 5462208 free, 283116 used, 2422152 buff/cache

There is indeed a problem, but this is an optimization by the allocator (probably glibc in your case) and unrelated to the library, as [**@nickaein**](https://github.com/nickaein) said.

If you add malloc\_trim(0) in your code:

while (!jsons.empty()) {

jsons.pop\_back();

}

+ malloc\_trim(0);

}

you will find everythink will be OK.  
In windows we can not reproduce because what we use is not glibc, i think.

Other Test:  
I had wrote other program to malloc many small memory with glibc, and the problem will still alive.My program is unrelated to the library, it just malloc and free many small memory.

Anyway, the probleam is unrelated to the library.  
If we add malloc\_trim(0) into the library, there will be many calls during the parsering, which will reduce performance.So the better solution is add malloc\_trim(0) in your code.

添加 malloc\_trim(0)来避免内存泄漏