
Subject: Re: Map implementation

Posted by [Novo](#) on Wed, 17 Apr 2019 14:13:01 GMT

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

mirek wrote on Wed, 17 April 2019 02:54

Well, nothing surprising there, right?

Yes, but ...

```
$ ./test_ht_perf
```

```
Mem used: 78128 Kb
```

```
Index<int> Add: 7.035
```

```
Index<int> Unlink: 12.272
```

```
Mem used: 2592740 Kb
```

```
Mem used after Sweep: 3800292 Kb
```

```
Mem used: 3769040 Kb
```

```
std::set<int> insert: 7.381
```

```
std::set<int> erase: 4.296
```

```
Mem used: 7603920 Kb
```

```
if (true) {
    Vector<Index<int> > v;
    v.SetCount(v_num);
    const int isize = 100;
    Cout() << "Mem used: " << MemoryUsedKb() - curMU << " Kb" << EOL;
    TimeStop ts;
    for (int i = 0; i < isize; ++i) {
        const int jsize = v_num;
        for (int j = 0; j < jsize; ++j)
            v[j].Add(i);
    }
    Cout() << "Index<int> Add: " << ts.ToString() << EOL;
    ts.Reset();
    for (int i = 0; i < isize; ++i) {
        const int jsize = v_num;
        for (int j = 0; j < jsize; ++j)
            v[j].UnlinkKey(i);
    }
    Cout() << "Index<int> Unlink: " << ts.ToString() << EOL;
    Cout() << "Mem used: " << MemoryUsedKb() - curMU << " Kb" << EOL;
    const int jsize = v_num;
    for (int j = 0; j < jsize; ++j)
        v[j].Sweep();
    Cout() << "Mem used after Sweep: " << MemoryUsedKb() - curMU << " Kb" << EOL;
}

if (true) {
```

```

std::set<int>* v;
v = new std::set<int>[v_num];
const int isize = 100;
Cout() << "Mem used: " << MemoryUsedKb() - curMU << " Kb" << EOL;
TimeStop ts;
for (int i = 0; i < isize; ++i) {
    const int jsize = v_num;
    for (int j = 0; j < jsize; ++j)
        v[j].insert(i);
}
Cout() << "std::set<int> insert: " << ts.ToString() << EOL;
ts.Reset();
for (int i = 0; i < isize; ++i) {
    const int jsize = v_num;
    for (int j = 0; j < jsize; ++j)
        v[j].erase(i);
}
Cout() << "std::set<int> erase: " << ts.ToString() << EOL;
Cout() << "Mem used: " << MemoryUsedKb() - curMU << " Kb" << EOL;
}

```

Project is attached.

std::set<int> erase is three times faster than Index<int> Unlink.

After calling Index::Sweep even more memory is used. I guess this is a problem with the allocator.

And Index invalidates pointers.

So ...

File Attachments

1) [test_ht_perf.zip](#), downloaded 313 times
