Subject: Re: Map implementation
Posted by mirek on Mon, 01 Jul 2019 15:53:08 GMT
View Forum Message <> Reply to Message

Things are quite different if instead of incremental pattern you feed in random data:

```
#include <Core/Core.h>
#include <set>

using namespace Upp;

CONSOLE_APP_MAIN
{
#ifdef _DEBUG
 const int v_num = 10000;
#else
 const int v_num = 1000;
#endif

 const int isize = 100;
 const int N = 100;

 Vector<int> data;
 for(int i = 0; i < isize * v_num; i++)
  data.Add(Random());

 for(int ii = 0; ii < N; ii++) {
  {
   Vector<Index<int> > v;
   v.SetCount(v_num);
   {
    RTIMING("inner FindAdd v_num");
    int *s = data;
    for (int i = 0; i < isize; ++i)
     for (int j = 0; j < v_num; ++j)
      v[j].FindAdd(*s++);
   }
   {
    RTIMING("inner UnlinkKey v_num");
    int *s = data;
    for (int i = 0; i < isize; ++i)
     for (int j = 0; j < v_num; ++j)
      v[j].UnlinkKey(*s++);
   }
   RTIMING("inner Sweep v_num");
   const int jsize = v_num;
   for (int j = 0; j < jsize; ++j)
```

```
  v[j].Sweep();
}
{
 Vector<Index<int> > v;
 v.SetCount(v_num);
 {
  RTIMING("outer FindAdd v_num");
  int *s = data;
  for (int j = 0; j < v_num; ++j)
   for (int i = 0; i < isize; ++i)
    v[j].FindAdd(*s++);
 }
 {
  RTIMING("outer UnlinkKey v_num");
  int *s = data;
  for (int j = 0; j < v_num; ++j)
   for (int i = 0; i < isize; ++i)
    v[j].UnlinkKey(*s++);
 }
 RTIMING("outer Sweep v_num");
 const int jsize = v_num;
 for (int j = 0; j < jsize; ++j)
  v[j].Sweep();
}

{
 std::set<int> *v = new std::set<int>[v_num];
 {
  RTIMING("outer insert v_num");
  int *s = data;
  for (int j = 0; j < v_num; ++j)
   for (int i = 0; i < isize; ++i)
    v[j].insert(*s++);
 }

 {
  RTIMING("outer erase v_num");
  int *s = data;
  for (int j = 0; j < v_num; ++j)
   for (int i = 0; i < isize; ++i)
    v[j].erase(*s++);
 }
 delete[] v;
}

{
 std::set<int> *v = new std::set<int>[v_num];
 {
```

```
    RTIMING("inner insert v_num");
    int *s = data;
    for (int i = 0; i < isize; ++i)
     for (int j = 0; j < v_num; ++j)
      v[j].insert(*s++);
   }

   {
    RTIMING("inner erase v_num");
    int *s = data;
    for (int i = 0; i < isize; ++i)
     for (int j = 0; j < v_num; ++j)
      v[j].erase(*s++);
   }
   delete[] v;
  }
 }
}
```

I guess incremental data somehow favors set, my guess is that it works as accidental prefetch here...
