CoCoALib - Slug #1557

Reading list of rationals is too slow

05 Jan 2021 21:43 - John Abbott

Status:	Closed	**Start date:**	05 Jan 2021
Priority:	Normal	**Due date:**	
Assignee:	John Abbott	**% Done:**	100%
Category:	Improving	**Estimated time:**	3.11 hours
Target version:	CoCoALib-0.99850	**Spent time:**	3.15 hours

Description

Winfried Bruns sent me a file with about 100000 rationals (total size about 130Mbytes).
If I read the file in using an obvious C++ loop (which stops when reading 0, which I added as an end marker), it takes about 8.8s
A modified version of the file represents the list as a CoCoA-5 comma-separated list: CoCoA-5 reads the list in about 4s.

Why is the direct C++ impl slower?

Investigate & fix.

History

#1 - 05 Jan 2021 21:55 - John Abbott

The profiler indicates that major costs are hgcd (from GMP) and ScanUnsignedIntegerLiteral (from CoCoALib).

JAA is quite surprised that ScanUnsignedIntegerLiteral is so costly... he will investigate.

#2 - 06 Jan 2021 11:32 - John Abbott

- Status changed from New to In Progress

- Assignee set to John Abbott

- % Done changed from 0 to 10

I'm still puzzled: ScanUnsignedIntegerLiteral continue to be slower than whatever the CoCoA-5 interpreter does; could it just be that reading (from an istream) chars 1 at a time is slow? The interpreter reads a whole line in one go, and then scans that.

The slow impl (about 8.8s) is this:

```
    // while (true)
    // {
    //   const char ch = in.peek(); // this may set eofbit
    //   if (!in.good() || !isdigit(ch)) break;
    //   in.ignore();
    //   digits += ch;
    // }
```

The faster impl (5.5s) is this: **[corrected 2021-01-07]**

```
    char ch;
    while (true)
    {
      in.get(ch);
      if (in.eof()) { in.clear(); break; }
      if (!isdigit(ch)) { in.unget(); break; }
      digits += ch;
    }
```

I also tried with back_inserter instead of op+= but that was a bit slower (5.8s).

#3 - 06 Jan 2021 11:40 - John Abbott

I have found a potentially useful entry on **StackOverflow**: link
https://stackoverflow.com/questions/9272276/can-you-specify-what-isnt-a-delimiter-in-stdgetline
To be honest, I am a little surprised that this is not already part of a standard library.

Michael Burr posted the following code (**I have not tried it**):

```
#include <functional>
#include <iostream>
#include <string>

using namespace std;

template <typename Predicate>
istream& getline_until( istream& is, string& str, Predicate pred)
{
    bool changed = false;
    istream::sentry k(is,true);

    if (bool(k)) {
        streambuf& rdbuf(*is.rdbuf());
        str.erase();

        istream::traits_type::int_type ch = rdbuf.sgetc(); // get next char, but don't move stream position
        for (;;ch = rdbuf.sgetc()) {
            if (istream::traits_type::eof() == ch) {
                is.setstate(ios_base::eofbit);
                break;
            }
            changed = true;
            rdbuf.sbumpc(); // move stream position to consume char
            if (pred(istream::traits_type::to_char_type(ch))) {
                break;
            }
            str.append(1,istream::traits_type::to_char_type(ch));
            if (str.size() == str.max_size()) {
                is.setstate(ios_base::failbit);
                break;
            }
        }

        if (!changed) {
            is.setstate(ios_base::failbit);
        }
    }

    return is;
}
```

#4 - 07 Jan 2021 20:02 - John Abbott

- Status changed from In Progress to Resolved

- % Done changed from 10 to 50

There was a bug in my first version of the faster code: in.get(ch) does not put EOF into ch when EOF is reached -- I had misread the manual. Took a long to track down the bug (because many tests had passed).

Should I try that code copied from StackOverflow? :-/

#5 - 10 Mar 2023 17:46 - John Abbott

- Status changed from Resolved to Closed

- % Done changed from 50 to 100

- Estimated time set to 3.11 h

This is not so important. Yes, it is strange that CoCoA-5 can read the input so fast... but it is not important.

#6 - 10 Mar 2023 18:23 - Anna Maria Bigatti

- Subject changed from Readling list of rationals is too slow to Reading list of rationals is too slow