

CoCoA-5 - Feature #504

New operators: += -= *= /=

02 Apr 2014 01:13 - John Abbott

Status:	In Progress	Start date:	02 Apr 2014
Priority:	Normal	Due date:	
Assignee:	John Abbott	% Done:	20%
Category:	CoCoA-5 function: new	Estimated time:	0.00 hour
Target version:	CoCoA-5.?.?	Spent time:	1.60 hour
Description			
<p>Now that I've implemented new fns incr and decr, I wonder whether it wouldn't have been more sensible to implement instead new operators +=, -=, *= and /= with the same semantics as in C++. They'd certainly be more useful than incr and decr (<i>e.g.</i> creating a new polynomial by summing to it one term at a time, though it'd be better to use a "geobucket" for that).</p> <p>For someone who knows neither, would it be harder to learn incr(ref counter); or counter += 1;? There is a slight difference: incr expects counter to contain an integer, but += should not impose any such limitation.</p> <p>Would it make sense to add these operators? Or perhaps just some of them? [since they're new operators, we'd have to change the parser/interpreter, but JAA does not think that would be too hard]</p>			
Related issues:			
Related to CoCoA-5 - Design #364: Incr command/function		Closed	03 Jun 2013
Related to CoCoA-5 - Feature #518: incr/decr functions: to be used in packages		Closed	03 Apr 2014

History

#1 - 02 Apr 2014 08:02 - Anna Maria Bigatti

- Target version set to CoCoA-5.1.0 Easter14
- % Done changed from 0 to 10

I think += is a big step over **incr**. After all how often we have **for** loops with step different from one?
I agree it would be nice, but just for a small elite.

#2 - 02 Apr 2014 11:39 - Christof Soeger

I like the += syntax very much, so I would vote for it.
For the incr, why not also ++?

#3 - 02 Apr 2014 12:15 - John Abbott

- Status changed from New to In Progress

We could also have the ++ operator but cannot have -- because that is used to mark end-of-line comments. In my opinion/experience, incrementing is a far more common operation than decrementing. However, a C++ programmer who wants to use CoCoA would probably be surprised if ++ works but -- does not.

Here are four potential code samples:

```
* (A) * ++counter;
* (B) * counter += 1;
* (C) * incr counter;
* (D) * incr(ref counter);
* (E) * counter := counter+1;
```

We can already do (D) and (E); I almost prefer (E) to (D) -- oops!
After any years of C/C++ programming I find (A) clear and concise, but could imagine someone without C/C++ experience could find it too terse/opaque. I do feel that (B) is a reasonable compromise.

#4 - 02 Apr 2014 12:21 - John Abbott

Would we really want all 4 operators +=, -=, *= and /=?

Probably /= would be used only rarely, but if we have the other 3 then for symmetry we should also have /=.

I have deliberately omitted %= . CoCoA-4 did have a % operator, but it is not defined in CoCoA-5 (as it is not needed) -- the interpreter recognises it but says that it is not (yet) defined. If ever % is defined, we can then consider adding %=.

#5 - 02 Apr 2014 13:37 - Christof Soeger

From the existing alternatives I also prefer (E) since it is clearly understandable. If you cannot have -- you should also not have ++, it is just to confusing and inconsistent. Having i += 1; is good enough.

I think you should have all 4 operators.

#6 - 02 Apr 2014 14:21 - Anna Maria Bigatti

Christof Soeger wrote:

If you cannot have -- you should also not have ++, it is just to confusing and inconsistent.

I agree with that: no (A)

I'm against (C) as well: I do not like commands, I prefer functions/procedures, so I quite like (D).

I'm mildly against (B) += in CoCoA-5 (while I like it a lot in C++)

#7 - 09 Apr 2014 17:43 - John Abbott

- Target version changed from CoCoA-5.1.0 Easter14 to CoCoA-5.1.1 Seoul14

#8 - 27 Aug 2014 18:10 - Anna Maria Bigatti

- Target version changed from CoCoA-5.1.1 Seoul14 to CoCoA-5.?.?

#9 - 25 Oct 2019 11:46 - John Abbott

- Assignee set to John Abbott

- % Done changed from 10 to 20

One big advantage of rejecting the idea of introducing += and friends is that there would be no need to fiddle around inside the interpreter... to my mind the code is not as self-documenting at the original author believes :-/

Earlier comments have effectively voted against (A) and (C); we already have (D) and (E). So the only point to resolve is (B). Christof was in favour (comment 2), Anna was against (comment 6).

The advantage of += and friends is that they are well-appreciated by C/C++ programmers.

A disadvantage is that people who do not know C/C++ might find them cryptic. Another point is that in CoCoA-5 they would probably not be especially efficient -- somehow in C++ they "feel like" they ought to be particularly efficient. Anyway, the aim of CoCoA-5 is not to be especially efficient (but also not to be needlessly inefficient).

It seems reasonable to assume that $\mathbf{A} += \mathbf{B};$ would mean exactly the same as $\mathbf{A} := \mathbf{A} + \mathbf{B}$ (with the exception that \mathbf{A} is evaluated only once). This could lead to some mildly surprising uses: let R be a ring and I an ideal in R , then $\mathbf{R} /= \mathbf{I}$ would mean $\mathbf{R} := \mathbf{R}/I$. Of course, no one is compelled to write such a command.

NOTE in the CoCoA-5 packages **incr** is used in about 35 places (currently). Not sure if I can find out how often we use something like $\mathbf{A} := \mathbf{A} + \mathbf{B};$

#10 - 25 Oct 2019 11:55 - John Abbott

Here is a command for finding lines which contain commands like $\mathbf{A} := \mathbf{A} + \text{blah}$ or some other operator -- it does find some other stuff as well (and does not find things like $L[k] := L[k] + \text{sthg}$)

```
egrep "(\\<\\w*\\>) *:= *\\<\\1\\>" *.cpkg5
```

With the current (2019-10-25) versions of the packages there are about 160 instances.

There are about 10 cases of the form $\mathbf{A} := \mathbf{A} + 1$ which could be replaced immediately by **incr(ref A)**.

I noted about 35 cases where I *might* be tempted to **op=** instead of $\mathbf{A} := \mathbf{A} \text{ op } \mathbf{B}$