
Subject: The age of multicore has just started for U++....

Posted by [mirek](#) on Sun, 08 Apr 2007 10:00:40 GMT

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

...well, perhaps only a little step, but I have just made experimental U++ website generation package, using the new "CoWork" class (in the next dev release) to use both of my E4300 cores to do the job, to became the "real thing" (removing "experimental" status). Means I will now use to do the "real job".

Reduces U++ website generation from 23s to 14s.

Mirek

Subject: Re: The age of multicore has just started for U++....

Posted by [exolon](#) on Sun, 08 Apr 2007 14:33:54 GMT

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

What does this Cowork class do, and how?

Subject: Re: The age of multicore has just started for U++....

Posted by [mirek](#) on Sun, 08 Apr 2007 15:14:59 GMT

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

exolon wrote on Sun, 08 April 2007 10:33What does this Cowork class do, and how?

Well, detailed description will come with new Core, but essentially it is a thread pool. A tool to simply paralellize loops.

E.g. this is some original loop from the upweb:

```
VectorMap<String, String> reflink;
```

```
struct ScanTopicIterator : RichText::Iterator {  
    String      link;
```

```
    virtual bool operator()(int pos, const RichPara& para)  
    {  
        if(!IsNull(para.format.label)) {  
            reflink.Add(para.format.label, link);  
        }  
        return false;  
    }  
};
```

```

void GatherRefLinks(const char *upp)
{
    Progress pi;
    pi.AlignText(ALIGN_LEFT);
    for(FindFile pff(AppendFileName(upp, "*.tpp")); pff; pff.Next()) {
        if(pff.IsFolder()) {
            pi.Step();
            String package = pff.GetName();
            String pdir = AppendFileName(upp, package);
            TopicLink tl;
            tl.package = package;
            for(FindFile ff(AppendFileName(pdir, "*.tpp")); ff; ff.Next()) {
                if(ff.IsFolder()) {
                    String group = GetFileTitle(ff.GetName());
                    tl.group = group;
                    String dir = AppendFileName(pdir, ff.GetName());
                    for(FindFile ft(AppendFileName(dir, "*.tpp")); ft; ft.Next()) {
                        if(ft.IsFile()) {
                            String path = AppendFileName(dir, ft.GetName());
                            tl.topic = GetFileTitle(ft.GetName());
                            String link = TopicLinkString(tl);
                            pi.SetText("Indexing topic " + tl.topic);
                            ScanTopicIterator sti;
                            sti.link = link;
                            ParseQTF(ReadTopic(LoadFile(path))).Iterate(sti);
                        }
                    }
                }
            }
        }
    }
}

```

parallel version with CoWork:

```

StaticCriticalSection    reflink_lock;
VectorMap<String, String> reflink;

struct ScanTopicIterator : RichText::Iterator {
    String    link;

    virtual bool operator()(int pos, const RichPara& para)
    {
        if(!IsNull(para.format.label)) {

```

```

INTERLOCKED_(reflink_lock)
reflink.Add(para.format.label, link);
}
return false;
}
};

```

```

static void sDoFile(const char *path, const char *link)
{
    ScanTopicIterator sti;
    sti.link = link;
    ParseQTF(ReadTopic(LoadFile(path))).Iterate(sti);
}

```

```

void GatherRefLinks(const char *upp)
{
    CoWork work;
    Progress pi;
    pi.AlignText(ALIGN_LEFT);
    for(FindFile pff(AppendFileName(upp, "*.tpp")); pff; pff.Next()) {
        if(pff.IsFolder()) {
            pi.Step();
            String package = pff.GetName();
            String pdir = AppendFileName(upp, package);
            TopicLink tl;
            tl.package = package;
            for(FindFile ff(AppendFileName(pdir, "*.tpp")); ff; ff.Next()) {
                if(ff.IsFolder()) {
                    String group = GetFileTitle(ff.GetName());
                    tl.group = group;
                    String dir = AppendFileName(pdir, ff.GetName());
                    for(FindFile ft(AppendFileName(dir, "*.tpp")); ft; ft.Next()) {
                        if(ft.IsFile()) {
                            String path = AppendFileName(dir, ft.GetName());
                            tl.topic = GetFileTitle(ft.GetName());
                            String link = TopicLinkString(tl);
                            pi.SetText("Indexing topic " + tl.topic);
                            work & callback2(sDoFile, path, link);
                        }
                    }
                }
            }
        }
    }
}

```

The advantage is that you do not have start/join/manage threads, in fact, CoWork starts a couple of threads ("thread pool") first time it is used and never quits them until application exists.

Mirek

Subject: Re: The age of multicore has just started for U++....

Posted by [exolon](#) on Sun, 08 Apr 2007 20:50:18 GMT

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

Nice... so it looks like an asynchronous function call using threads?

Subject: Re: The age of multicore has just started for U++....

Posted by [Novo](#) on Wed, 11 Apr 2007 03:23:22 GMT

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

Sorry. I do not want to be boring, but isn't easier to use specialized libraries for this purpose? Something like ACE (<http://www.cs.wustl.edu/~schmidt/ACE-overview.html>).

Those guys are 15 years ahead of you.

Subject: Re: The age of multicore has just started for U++....

Posted by [mirek](#) on Wed, 11 Apr 2007 08:35:15 GMT

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

Novo wrote on Tue, 10 April 2007 23:23 Sorry. I do not want to be boring, but isn't easier to use specialized libraries for this purpose? Something like ACE (<http://www.cs.wustl.edu/~schmidt/ACE-overview.html>).

Those guys are 15 years ahead of you.

Maybe. How does the above rewrite look like in ACE? (I was unable to find anything about thread pool in docs).

Moreover, the thread pool patter is well known. However, the aim of CoWork is to provide interface/implementation that exploits the rest of U++ library, like callbacks (perfect fit here btw).

Mirek

Subject: Re: The age of multicore has just started for U++....

Posted by [zsolt](#) on Wed, 11 Apr 2007 18:09:26 GMT

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

Novo wrote on Wed, 11 April 2007 05:23 Sorry. I do not want to be boring, but isn't easier to use specialized libraries for this purpose? Something like ACE (<http://www.cs.wustl.edu/~schmidt/ACE-overview.html>).

Those guys are 15 years ahead of you.

I tried ACE some years ago, but it is a very oldschool stuff, compared to U++, I think.

Subject: Re: The age of multicore has just started for U++....

Posted by [okigan](#) on Fri, 13 Apr 2007 00:41:35 GMT

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

ACE is very heavy dependency...

Why the peculiar choice of & operator overloading?

-Okigan

Subject: Re: The age of multicore has just started for U++....

Posted by [mirek](#) on Fri, 13 Apr 2007 06:23:44 GMT

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

okigan wrote on Thu, 12 April 2007 20:41 ACE is very heavy dependency...

Why the peculiar choice of & operator overloading?

-Okigan

Please, consider this one "highly experimental" (and questionable), but IMO it resembles ending linux console line with '&'....

Of course, simple method named "Do" would do to

Mirek

Subject: Re: The age of multicore has just started for U++....

Posted by [kenhty](#) on Sat, 18 Aug 2007 05:36:25 GMT

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

Anyone consider using TBB (<http://osstbb.intel.com/>)

Subject: Re: The age of multicore has just started for U++....

Posted by [mirek](#) on Sat, 18 Aug 2007 08:05:57 GMT

kenhty wrote on Sat, 18 August 2007 01:36 Anyone consider using TBB (<http://osstbb.intel.com/>)

I have noticed its existence.

I have looked at the example of `parallel_for` and at the first glance, it looks as inferior solution to me... A lot of code to write, compared to CoWork.... (I might be wrong... I will perhaps try to reimplement the example given by Intel with CoWork).
