

Re: What OpenMP brings to C++?

Source: <http://www.tech-archive.net/Archive/VC/microsoft.public.vc.language/2004-11/0296.html>

From: Simon Trew (*noneofyour_at_business.guv*)

Date: 11/07/04

Date: Sun, 7 Nov 2004 08:00:58 -0000

"Vladimir_petter" <vladp72@hotmail.com> wrote in message
news:Oq#nqxIxEHA.2316@TK2MSFTNGP15.phx.gbl...

> *I can see advantages of using Open MP for "parallel programming" (cause
C++
> language does not give us any othe way to express this), but I kind of
doubt
> that it is good for concurrent programming.*

It's providing an important feature that, as far as I know, has only been a
language construct in one language: Occam (for transputers). There, every
compound statement was either PAR or SEQ (IIRC). PAR meant the statements
inside it could be executed in parallel, SEQ that they must be executed in
series. One could nest PAR statements inside SEQ ones and vice versa

> *I can see that this loop can be great deal optimized using parallel
> programming (on Itanium), but making this loops parallel using concurrent
> programming just does not make sense to me. It simply too expensive to
> delegate a single x86 (or x64) "add" instruction in a separate thread
.[snip]
> That is one of the things that bothers me a lot. In all OpenMP peppers
I've
> read so far there are examples like that do not make sense to parallelize
in
> concurrent program. It is just a waste of CPU.*

It's a (big) hint to the optimizing comiler. If the compiler decides that
parallelization isn't worth it, it won't do it. (In an ideal world...)

> *In that case the question is what are the criteria that would help us to
> decide which way to choose for a particular problem.
>
> In particularly I do not see Open MP overlap nicely with idea of (IOCP)
> thread pool.
>
> Vladimir.*

You're right, threads are relatively large-scale things when compared to the

microsoft.public.vc.language: Re: What OpenMP brings to C++?

parallelizing of loops etc. And I think this is the essence of the answer.

Small-scale parallelization is done by coding in OpenMP or Occam's PAR/SEQ construct; larger-scale parallelization using threading, multitasking, DCOM, CORBA etc.

S.