Subject: CoWork::FinLock
Posted by mirek on Sat, 09 Jan 2016 19:24:59 GMT
View Forum Message <> Reply to Message

Do not ask me why, but recently I was having fun with parallel programming (I looks like it will be unintended addition for the next release... :).

In the process, while optimizing everything around CoWork, I got an interesting idea, CoWork::FinLock. It is intended to lock storing partial results into shared target at the end of scheduled routine. The idea is that CoWork has to lock some mutex anyway after the routine ends, so it is possible to 'prelock' this internal mutex and join both critical sections.

Usage looks something like this:

```
template <typename I, typename Lambda>
void CoLoop(I begin, I end, const Lambda& lambda)
{
 size_t chunk = max(size_t((end - begin) / CPU_Cores()), (size_t)1);
 CoWork co;
 while(begin < end) {
  co & [=] {
   lambda(begin, begin + min(chunk, size_t(end - begin)));
  };
  begin += chunk;
 }
}

template <class I, class Better>
I CoBest(I begin, I end, const Better& better)
{
 I best = begin++;
 CoLoop(begin, end,
  [=, &best](I i, I e) {
   I b = i++;
   while(i < e) {
    if(better(*i, *b))
     b = i;
    i++;
   }
   CoWork::FinLock(); // better is shared across threads, needs a lock!
   if(better(*b, *best))
    best = b;
  }
 );
 return best;
}
```

(CoBest is intended to lookup e.g. minimum or maximum, 'best' value).