Script–Fu

Friday 22 January 2010 07:04

Trowel is written around AnyEvent, an event loop programming framework. That means the program is devided in units that can continue to run without waiting for each other. This is convenient when we're using network connections that are apt to hang – like twitter – so the parts that have nothing to do with the network connection can keep going independent of the state of the network connection (in theory). The downside is that you never know in what order the various units interact and to prevent race conditions in which two or more units try to change the same central data at the same time the interaction has to be atomic. For Trowel this means one unit pushes the collected tweets onto an array while the other unit pops them of to display them, so at my level of abstraction the units don't have to explicitly wait for each other to finish changing the contents of the array. I assume there's one processor that is changing values in memory and in one processor cycle the push and the pop can't be done simultaneously.

With linux on Cell Broadband Engine hardware it's possible to do awesome things. But the trick is that the Cell consists of a rather slow processor and about 6 super fast specialised processor units. To create programs that utilize all the processors it is required to write them about the same as with an event loop. The program would require multiple threads that try to keep all the specialized processors busy, and the hard part is to manage the data that is used in all those threads. because if 6 units want to pop or push the same array at the same time you're suddenly looking at one of those that will have to wait up to 5 cycles, and a bigger chance that it should be affected by what the other 5 units have been doing.

Getting bored of some repetitive task in the Gimp I decided to write a script that would do the same task with the push of a button. The challange was that it had to be done in Scheme. Which is a pain. Just like with the Cell you're confronted with how the underlaying hardware really works and not how your brain works. After a while your brain starts to work like the hardware though and it gets easier, but it shows that writing optimized code is still hard. And why a lot of code is still slow on today's blazingly fast hardware...

by Roland van Ipenburg http://www.xs4all.nl/~ipenburg/blog/posts/play/2010/01/22/script–fu/