

Re: RAM-CPU Singularity

Source: <http://coding.derkeiler.com/Archive/General/comp.theory/2006-02/msg00026.html>

- *From:* Karl Klose <kklose@xx>
 - *Date:* Thu, 09 Feb 2006 12:59:56 +0100
-

Paul E. Black schrieb:

On Tue, 07 Feb 2006 05:07:35 -0800, yaoziyua wrote:

... I have this idea of RAM-CPU singularity:

That is to say, software applications should sometimes treat the RAM and CPU resources as one single computing resource, using an idle resource to compensate the urgent needs of another kind of resource.

Based on space-time tradeoff principles in algorithm design theory, we can redesign object libraries like STL to be able to adapt to any extremely unbalanced space-time requirements (e.g. a very fast CPU but very limited RAM, or a very slow CPU but very rich RAM) and able to re-adapt to a new requirement on-the-fly. Application frameworks like MFC, VCL or GTK should also be redesigned to serve this purpose. But the details of resource management should be transparent to the end developer.

An intriguing idea! I don't know if the trade-off could be made at a fine grain, like page swapping, but it might work at a coarser grain.

Some computations "memoize" intermediate computations. If memory is tight, it may be better to recompute more things and memoize fewer.

I think aspect-oriented techniques could be an elegant way to achieve this variability of caching. On slow systems, one could deploy an aspect that caches all (or some; perhaps depending on speed/RAM-ratio) return values of calculations.

Another point would be to offer more abstract algorithm and data structures: instead of letting the developer choose the concrete algorithm or structure, he should be able to give hints on memory-usage (size) and usage frequency of the data. The library could then choose an appropriate algorithm.

These trade-off could be made at initial run time. That is, the program's start-up detects whether the execution environment is

Re: RAM-CPU Singularity

memory-rich or computation-rich, then run the branch/version optimized for that.

-paul-

In my scenario, the environment would choose an algorithm/data-structure factory and possibly deploy caching strategy aspects at startup.

Regards,
Karl

.