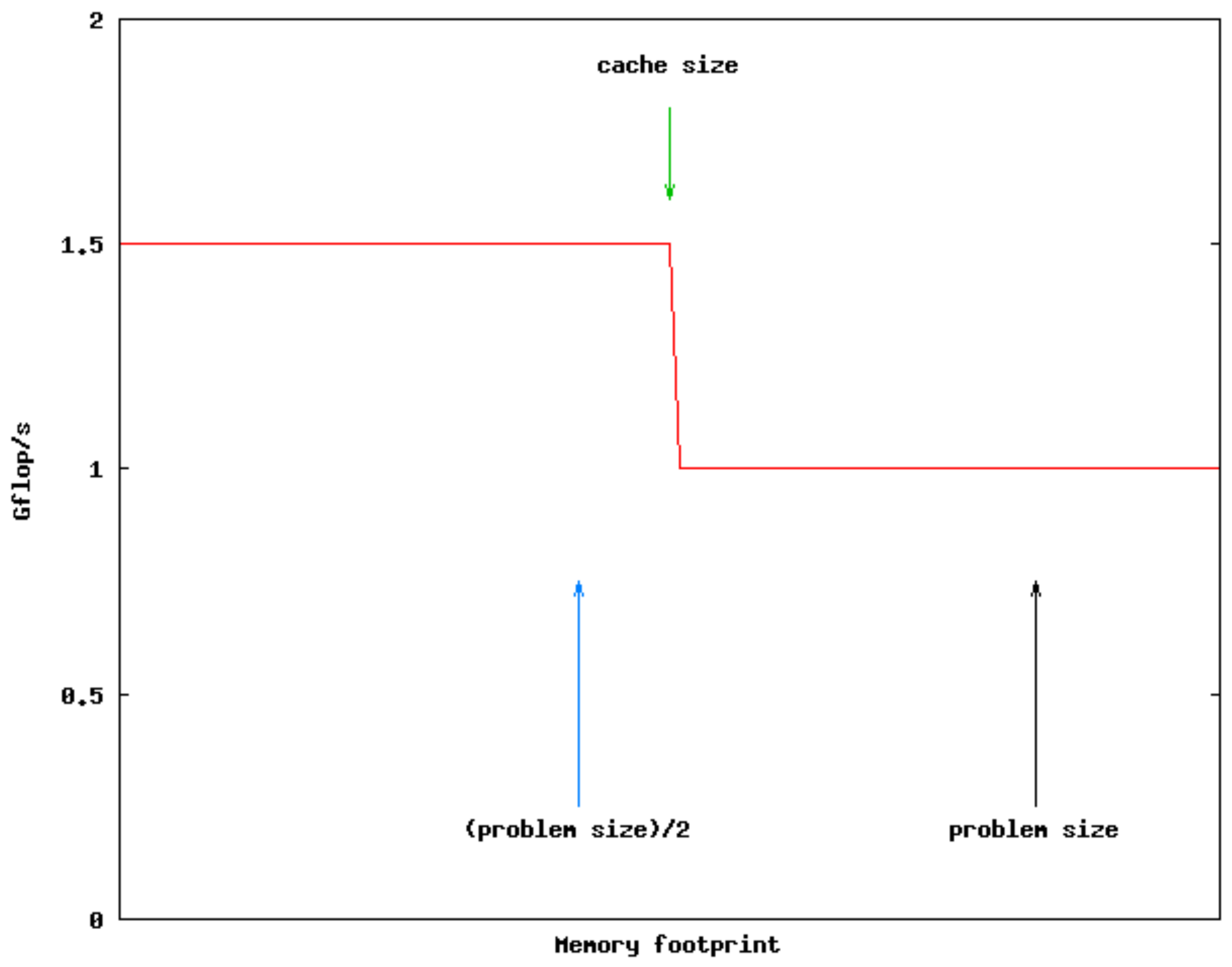
Superlinear speed-up

Some of you have observed a speed-up, that exceeded the expected upper limit, i.e. linear speed-up. How could this happen? Didn't we say, that Amdahl's law gives us an upper limit of the expected speed-up, and for a 100% parallel application, linear speed-up was the maximum?

If you observe speed-up above linear speed-up, there is nothing wrong! This can happen, and it can be also explained easily. Let's assume the following performance behaviour of a problem:



If we execute the code on a single thread, then we run it in the range where the performance is 1 Gflop/s. If it takes T(1) = 10 secs to run the code, then we have carried out 10 Gflops during the calculation.

If we now run the problem on 2 threads, we in principle divide the problem size by two, so each thread has a memory footprint of 1/2 of the problem size, and each has to execute 5 Gflops. With a memory footprint of this size, we move into a region where the reduced size can fit into one of our caches again, indicated by the green arrow, and the performance in this region is 1.5 Gflop/s. What does this mean for the execution time on 2 threads?

Let's do the math:

T(2)= 5Gflop / 1.5Gflop/s = 3.33s

and for the speed-up:

S(2)= T(1) /T (2) = 10s / 3.33s = 3

which is larger than the factor two, we should expect for linear speed-up!

Conclusion: it is in fact possible to get speed-ups above linear speed-up, and this is called 'superlinear' speed-up. In our formulation of Amdahl's law, those cache effects were not taken into account, as we also did not take other effects like parallel overhead cost, etc into account. As you might observe, this extra speed-up will usually cancel out with those other costs, if you increase the number of threads, and the speed-up will fall below linear speed-up again.