An Exerpt from
http://kevinclosson.wordpress.com/2009/05/14/you-buy-a-numa-system-oracle-says-disable-numa-what-gives-part-ii/

NUMA system fall into one three camps:

1. **Pioneer, Proprietary NUMA Implementations (PPNI).**
2. **Modern, Proprietary NUMA Implementations (MPNI).**
3. **Commodity NUMA Implementations (CNI).**

1) **Pioneer, Proprietary NUMA Implementations (PPNI).** The first commercial cache-coherent NUMA system was the Sequent NUMAQ-2000. Within a couple of years of that hardware release there were several other pioneer implementations brought to market by DG, DEC, SGI, Unisys and others. The implementation details of these pioneer NUMA systems varied hugely (e.g., interconnect technology, levels of OS NUMA awareness, etc). One thing these pioneer implementations all shared in common was the fact that they suffered **huge** ratios between local and remote memory latency. When I say huge, I'm talking as much as 50 to 1 for highly contended multiple-hop remote memory. The only reason these pioneer systems were brought to market was because they offered tremendous advancements in system bandwidth. The cost, however, was lumpy memory and thus software NUMA-awareness was of utmost importance. I would consider systems like the Sun E25K to be "second-generation" pioneer systems. Sure, the E25K suffered memory locality costs, but not as badly as the true pioneer systems. Few would argue that even the "second-generation" pioneer systems relied heavily on software NUMA-awareness.

2) **Modern, Proprietary NUMA Implementations (MPNI).** I'm not going to cite many systems here as cases in point. I don't aim

to wound the tender sensibilities of any hardware supplier. I can define what I mean by MPNI by simply stating that MPNI systems differ from PPNI in terms of remote to local memory latency ratios. In short, MPNI systems have very favorable L:R latency ratios. By very favorable, I mean significantly less than 2 to 1. An example of an MPNI system would be the Sun SPARC Enterprise M9000 Server which, according to my good friend Glenn Fawcett sports an approximate local to remote latency ratio of 1.3:1. In my opinion, it is not worth the complexities necessary to do proper, effective software NUMA awareness when there is only 30% disparity between the local and remote memory (at least not Oracle NUMA-awareness). Now, having said that, I know the M9000 supports scaling to multiple cabinets. I don't know enough about the crossbar (Juniper Interconnect) to say whether it requires any "hop" overhead in a multiple-cabinet configuration. Sun literature states point-to-point without caveats so the L:R ratio might remain constant as one adds cabinets. Nonetheless, the point being made here is that there exist today modern, proprietary NUMA implementations and concerns over Oracle NUMA awareness should be weighed according to MPNI capabilities—not arcane, PPNI capabilities.

3) **Commodity NUMA Implementations (CNI).** I don't feel compelled to hide my exuberance for modern NUMA implementations such as Intel QuickPath Interconnect and the HyperTransport (HT) used by AMD. The points I want to make about CNI are as follows:

- Memory Latency Ratios. While I've not stayed as up to speed on local-remote ratios with HT 3.0, I know that the Intel QPI-based systems offer very pleasant L:R ratios (e.g., 1.4:1 or better). More importantly, I should point out that even remote memory references in Nehalem-based Servers (Xeon 5500) are faster than all memory references in the previous generation Xeon-based systems (e.g., "Harpertown" Xeon 5400)!

- BIOS-Enabled NUMA. Commodity NUMA systems support the concept of boot-time NUMA enablement. When booted with NUMA disabled at the BIOS, the resultant memory architecture is commonly referred to as Sufficiently Uniform Memory Access (SUMA) or Sufficiently Uniform Memory Organization (SUMO).