Title:

Sanskrit based on-board super computer

Today’s limitation:

Already resolved (in Sanskrit grammar) "Inception vs perception" problem, end of 1850s' black body radiation problem and 1950s' halting problem are the same.

Halting problem can, in practice, be understood by the failure to avoid queue in parallelism and the inability of goal oriented selection using machine learning. This affects quantum computers also.

Details:

The role of black body radiation studies in the end of 1850s drived progress in modern physics, due to

initial difficulties in explaining the observed phenomena. The Heisenberg Uncertainty Principle, which states that one cannot accurately measure both the position and speed of a particle simultaneously, has led to conventional quantum theories like the "bootstrap theory" with still unsolved equations.

John Von Neumann, the physicist who made significant contributions to computer architecture and programming, offered key insights into the Halting Problem. This problem, along with other aspects of quantum physics as per Von Neumann, Weber, and others, is a part of several open-ended questions, even relating to consciousness.

Quantum computers aim to address the perfect parallelism from a unique perspective, but they also face limitations rooted in this problem. While I lack expertise in the technicalities of quantum computers and seek assistance from physicists for

specific questions, I offer to summarize the disputed and unproven aspects of quantum computers in both theory and application as follows:

1. Hadrons, as self-creating particles, present challenges, and data from the Large Hadron Collider (LHC) is inconclusive.
2. Existing practical applications like laser technology and quantum cryptography rely on 'conventional' quantum physics, despite unsolvable equations.
3. The practical use of quantum computers in theory is still several decades away, regardless of long-running projects. They primarily compete against traditional computers in relatively narrow domains, although the scope is expanding.
4. Quantum computers are unlikely to have a suitable form factor for onboard integration.
5. Cost and energy considerations pose significant challenges.

My intention of following questions is to stimulate thought and encourage contemplation.

Question 1: I humbly request to see if there is any practical example of a parallel program, language, or software/hardware architecture that operates efficiently without relying on queue. The utilization of queue impose limitations on system parallelism and also demonstrate shortcoming in the algorithm.

Question 2: Could you please provide evidence of the functionality and effectiveness of goal-oriented selection using machine learning? I appreciate any context or resources you can provide, such as the article available at https://ncbi.nlm.nih.gov/pmc/articles/PMC4186236/

For 1st question, examine these:

Bitcoin's no queue causing reliability issue; Uber-like apps' billing delays due to queuing; Memory limits in queue causing message loss in message queues; Databases recommend disabling OpLocks at file write caching level to address inconsistencies; Async protocols for WAN performance utilizing hidden queues; Mutexe implementations using queues; Itanium's failure for compiler-handled concurrency; Erlang parallelism used in Whatsapp being limited by queue memory; asynchronous circuits being limited by memory of the queue (analogous to queue is different here); concurrency handling issues in Domain Specific Languages; inevitability to increase out-of-order queue size in practice for super-scalar processor designs; queuing limits in real applications with GPU programming, and the list can go on and on.

"As far as the laws of mathematics refer to reality, they are not certain;

and as far as they are certain, they do not refer to reality" - Einstein

Humbly request utile real-world examples to challenge my views.

In all modesty, the complexity of the second question goes beyond this relatively short format. To summarize: Skinner's work with pigeons and boxes, James Olds' rat experiments with those boxes, and the discovery of extraordinary pleasure centers (referred to as 'chakras' in yoga) that override basic animal instincts; triggering of unethical human brain experiments thereafter prompting a comprehensive ban; the 2000 Nobel Prize-winning discovery of non-invasive brain study techniques rekindling interest in brain research; resulting 2003 'BRAIN Initiative' and the concurrent rise of machine learning.

Many traditional Indian philosophers have deliberated on goal-oriented

selection, opening up another complex discussion layer.

Solution:

Modern computers heavily rely on extensive optimizations at all levels to ensure usability. Systematic optimization is crucial to sustain progress and avoid issues like the problem of ending strings with zero, which persists across various levels of computing (https://queue.acm.org/detail.cfm?id=2010365). Notably, a grammar rule in Sanskrit prohibits statements from ending with zero or "sunya."

By leveraging Sanskrit, we harness time-tested optimizations ingrained in the grammar itself, derived from the few fundamental Maheswara sutras. Utilizing Panini's Sanskrit grammar rules and mimansa, I believe it is possible to construct a hardware

processor and directly employ Sanskrit as a programming language. This approach facilitates the creation of a supercomputing system that excels in all aspects.

Building upon existing research, my humble approach expands and enhances the context of Sanskrit in computer applications. I aim to demonstrate the following: In mimansa, there is a well-known situation where a servant, invited by a non-friend of the master during an ongoing lawsuit, encounters the person on the road. The master then cryptically says, "eat the poison and die," meaning they intend to change the servant's mind. Traditional computers, regardless of model or programming, cannot guarantee reaching this conclusion. However, with Sanskrit, we can confidently arrive at the precise conclusion, demonstrating the steps on paper in a concise manner.

Importantly, all necessary texts on

these subjects remain intact and available to this day, showcasing their enduring value over time.

Specifically, Sanskrit's "Viswamitra rule" solves "inception vs perception" and the implications exceed common understanding.

Supplementary Notes - 1:

The omission of Alan Turing's interpretation is due to the intricate discussion on Mimansa (particularly Uttara Mimansa) his interpretation would necessitate. Given the challenge of simplifying Mimansa, I've respectfully sidestepped it.

Supplementary Notes - 2:

The connection between physics and computer science becomes more apparent when examining the field of regex

indexing.

Supplementary Notes - 3:

The series example discussing the concept of infinity value, as illustrated in the Ramanujan Summation (https://www.cantorsparadise.com/the-ramanujan-summation-1-2-3-1-12-a8cc23dea793) and applied in string theory, can effectively be challenged by applying Adi Sankara's interpretation of 'rajju sarpa nyaya' and closely following Panini's 'vishwamitra sutra' grammar rule.

Supplementary Notes - 4:

To provide a glimpse into a specific discovery I made, let's think about this:
2 + 3
2 x 3

Envision an operator like '*' that signifies combining, '+' and 'x'.

2 * 3 = 1.

This might seem quite unconventional, yet I can validate it. I've encountered at least four pieces of work that reference this in the most enigmatic way conceivable.

1. Nakshatra maala stotra by Adi Sankaracharya
2. Bhagavata by Potana: The verses describing Krishna's leelas, where the narration was reported to Yashoda.
3. SaiSannidhi by Sri Acharya Bharadwaja
4. Panini's grammar.

As I delve deeper, these findings will undoubtedly lead to unprecedented insights about numbers that likely surpass any current imaginations. It's worth noting that this exploration will inevitably touch upon the realm of

cryptography.

Revision History:

2019Aug17 + Bangalore: Initial version
2023May25 + Bangalore: Current version