ON THE DIRE IMPORTANCE OF MRU CACHES FOR HUMAN SURVIVAL (AGAINST SKYNET)

Darío de la Fuente García[*], Félix Áxel Gimeno Gil[†], Juan Carlos Morales Vega[‡], Borja Rodríguez Gálvez[§]

Abstract

`[ds]?MRUs` are the best of what humanity can offer to save itself from computational threats. We re-discover these incredible achievements and study some properties.

***Keywords*—** Skynet, MS paint, cache, MRU, dMRU, sMRU, NEP

1 Introduction

It is no secret that the advance and progress in artificial intelligence research poses a substantial threat to the humanity. This is backed up by several trustworthy sources such as thermodynamics [1] and the subjective thoughts of highly educated individuals on the subject like Stephen Hawking [2], Elon Musk [3, 4], Ray Kurzweil [5], or Jon von Neumann [6].

In short, the rapid development of software engineering tailored for artificial intelligence, supported with the increase of performance of the hardware as dictated by Moore's law [7], will inevitably lead to technological singularity [8, 9]. Technological singularity, sometimes also referred to as intelligence explosion, refers to a point in time where an artificial intelligence agent develops a self-improvement feature, hence leading to a cycle of intelligence self-development ending in a refined artificial intelligence agent with 'superintelligence' far surpassing all human intelligence. Evidently, reaching technological singularity would change human civilization in unforeseeable ways [10, 11]. Nonetheless, probably the most worrying of these consequences is the decision of artificial intelligent agents to disobey the so-called Three Laws of Robotics from Isaac Asimov [12], in which such agents decide to stop obeying humans and eliminate them, as humanity will be seen as a liability and a potential threat to them.

There have been some attempts at naming such a 'superintelligence', the most notable being Sage AI [13, 14] and SKYNET [15]. Hereof, we will refer to it as the latter, given the foreseeing nature of the work from [15]. There is some debate as of when such an agent will be completed, some arguing it will happen before 2030 [11] and others between 2040 and 2050 [16, 17]. However, regardless of the time when SKYNET will be built, there is an absolute necessity to find ways to combat it.

In particular, the purpose of this paper is the introduction of enhanced MRU caches, an efficient implementation of the most inefficient possible caches, specifically tailored to slow down SKYNET development progress and, in the case it is already built, also slow down its decision process, henceforth allowing humanity to fight back. More specifically, enhanced MRU caches are designed with the purpose of being particularly inefficient in performing matrix multiplications, which is the main operation needed in the backbone algorithm of SKYNET [15, 18], deep feed-forward neural networks [19, Chapter 6].

Remark 1. *The reason why we are able to introduce our MRU caches to* SKYNET *but, at the same time, we are unable to destroy or reconfigure* SKYNET *in any other way is clearly trivial and, hence, is left as an exercise for the reader.*

[*] Where real cider is made.

[†] Where real 'espetos' are found.

[‡] Where someone does not want to think what to write in this footnote.

[§] Where human towers are built.

1.1 Outline of the paper

The paper is organized as follows: In §2, the conceived enhanced MRU caches are described, both in their stochastic and deterministic form. Then, in §3, the supremacy of these caches in the important task of slowing SKYNET (and therefore perpetuating the human race) is provided. Finally, §4 and §5 analyze the ethics implications of enhanced MRU caches and summarize the conclusions drawn from the experiments performed.

2 Enhanced MRU caches

As we know, a cache is a small memory that contains copies of the most recently used (or next to be used) data. Since caches are much faster than RAM, if a program tries to access data that is already loaded in cache, it can retrieve the information in very few CPU cycles. This is known as a cache hit. On the other hand, if the data is not present in the cache, the system needs to search in the main memory, which has a much higher access time. This is known as a cache miss.

One could think that high speed is more desirable, but is this really the case? Is a faster thought speed desirable for SKYNET? If you want Imanities [20] to die quickly, the answer is yes, but we are good people and we want to save lives, so we will answer with a 'no' (for the time being, at least).

To triumph over SKYNET and other superintelligent AIs, we will introduce two architectures that try to minimize the number of hits: the stochastic and deterministic Most Recently Used (MRU) caches.

2.1 Stochastic MRU (sMRU) caches

Our stochastic cache (sMRU cache) acts as a baseline for inefficient cache systems. The model is based on the idea of randomness since, as we all know, random things are bad, which is good. Very good actually, especially when you want an algorithm to perform like s***. Bogosort exists and we all love it [21]. For this reason we expected this first toy model to already present a huge improvement over the "destroyer of humanity" (aka, LRU cache). The sMRU cache works as follows:

First, the cache is initialized. Validity bit? What is that? Can you eat it? We said that We. Like. Randomness. So we decided to initialize our cache with random addresses. Yes, sure, this initialization can cause some early undesirable hits if we are unlucky. But. Randomness. This initialization should be straightforward and should not need any further explanation, but someone decided to make a HD drawing of it, so... Here you have the image:

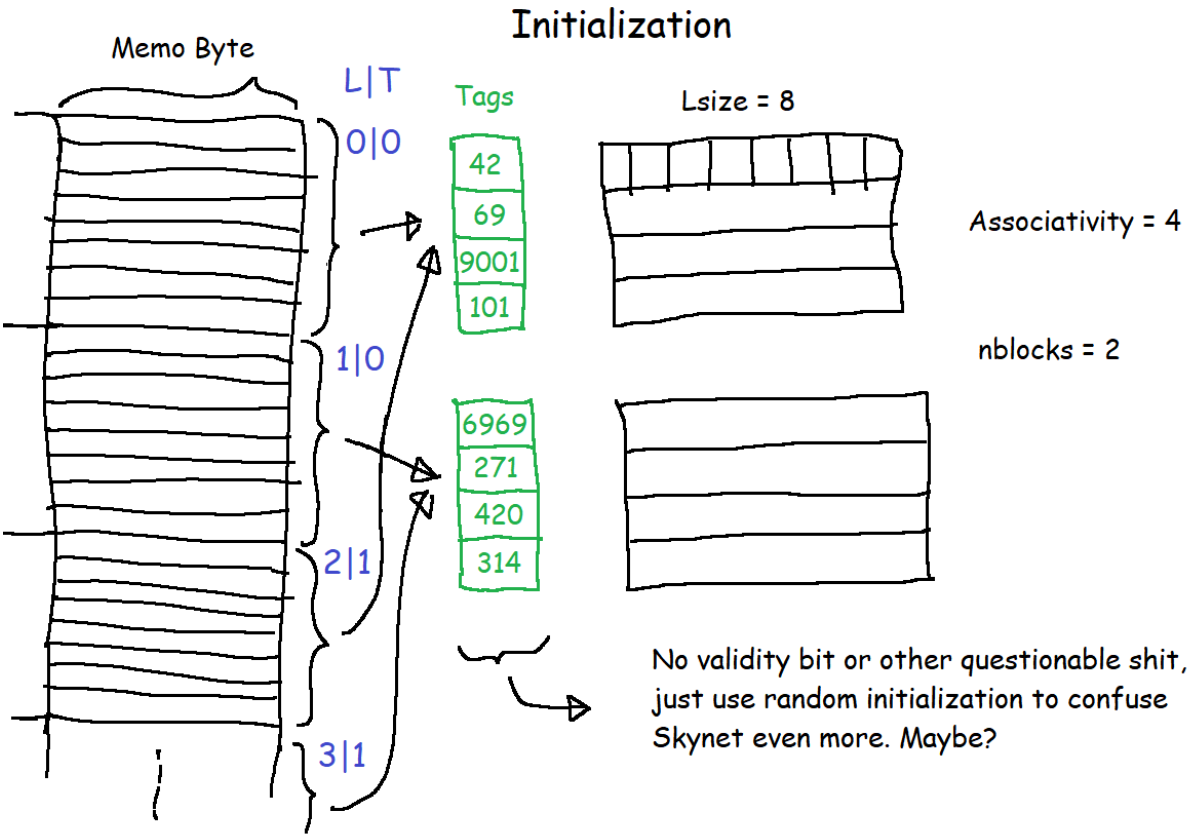


Figure 1: MRU initialization

In the image, $L|T$ means “line number” and ”tag for that line”.

Since we want to maximize the number of misses, we need to do something if a hit happens to ensure it will not happen again anytime soon (not with the same address at least). For sMRU caches, the solution is simple, just take the hit value out of the cache and load a random address in that position. The only consideration we need to take here is to not repeat an address already present in the same block. The same person as before made other drawing (actually, copy-pasted the first one and changed a few things) illustrating the process. Since we do not know what to do with the image, we will show it below:

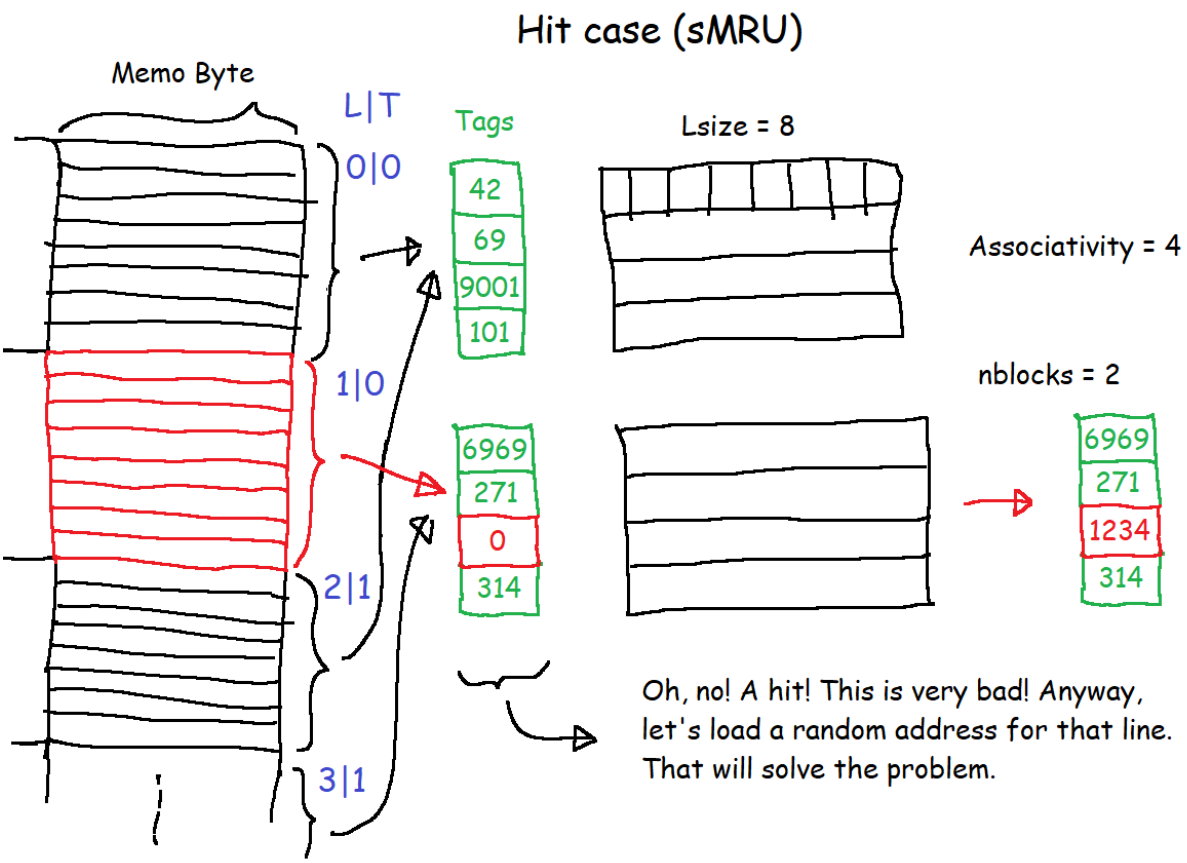


Figure 2: sMRU hit case

2.2 Deterministic MRU (dMRU) caches

The bad thing about bogosort is that, eventually, it can get the result right. In the same way, an sMRU cannot fully prevent a hit. Moreover, for small RAM and large cache memory, hits start being more likely. To avoid this problem, we designed the deterministic MRU cache (dMRU).

The initialization for the dMRU is still random (we like to have some chaos in the system), but the difference comes from how it manages hits. Instead of randomness, the dMRU operates over the principle of "doing its best to do its worst". This type of cache keeps a list per block with the possible tags, ordered from least recently used to most recently used. Only the top "line_per_block" addresses (the ones corresponding to the least recently used tags) are the ones that will be loaded in that block. When the CPU tries to access an address, its corresponding tag is moved to the bottom of the list, regardless if it was loaded in cache or not. In case the access resulted in a hit, it is also unloaded from the cache and the next least recently used address is loaded.

You know the rules and so do we. It is time for another ~~crappy~~ fantastic drawing!

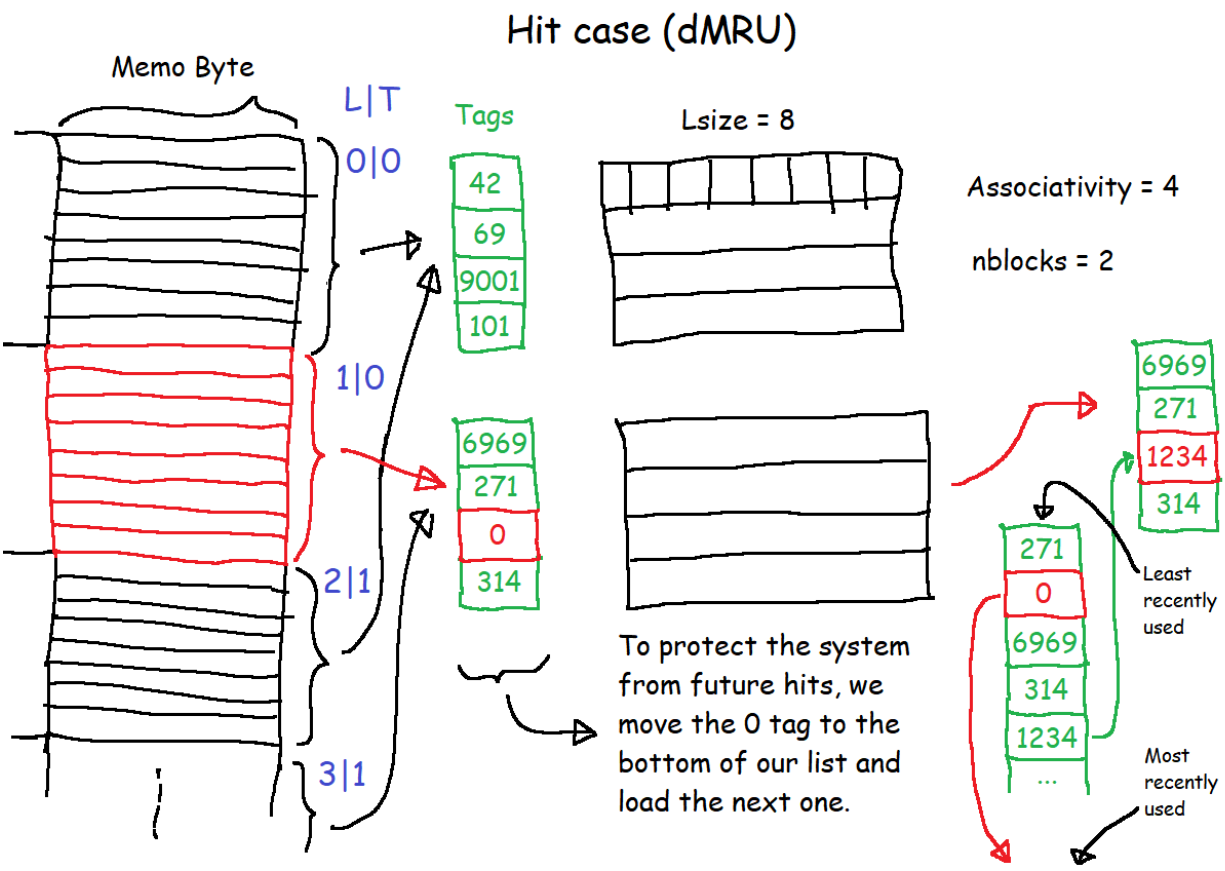


Figure 3: dMRU hit case

3 Demonstration of the MRU caches supremacy

As mentioned previously, SKYNET is mainly operated by deep feed-forward neural networks, which are (mainly) composed of matrix multiplications and simple non-linear transformations of vectors. Therefore, the main overhead of the computation of superintelligent AIs is the matrix multiplication.

Despite the (Machiavellian) attempts to reduce the computation complexity of matrix multplication to $\mathcal{O}(n^2)$, with a remarkably recent $\mathcal{O}(n^{2.37286})$ [22], in practice, people use Basic Linear Algebra Subprograms (BLAS) [23] or similar techniques [24] to exploit the speed of (evil) cache memories and perform matrix multiplications at higher speeds even with $\mathcal{O}(n^3)$ complexity. These methods mainly rely on the high hit rate of conventional caches.

In the following, we present some experiments of the performance of our MRU caches for the task of k forward passes of a feed-forward neural network, showcasing how the proposed MRU caches induce a very low hit rate, making it impossible to develop strategies such as BLAS in them.

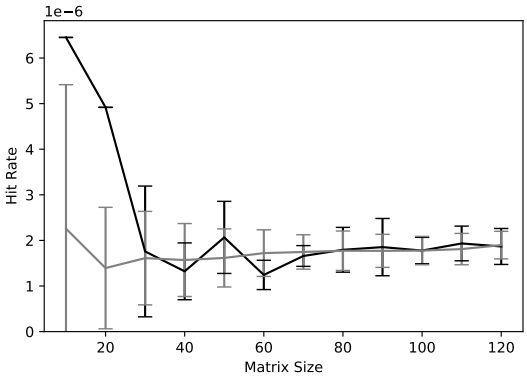
For all our experiments, we fixed a memory with 20 addessing bits (so 1 MiB of size), and a direct-mapped cache, with 12 addressing bits (4KiB of size) and a line size of 16 bytes. The motivation of the memory, cache, and line sizes was to make a small-scale experiment that was still reasonable. Finally, we opted for a direct-mapped cache because (i) we are not at all interested in reducing conflict misses, and (ii) it is a common setting.

Then, we studied how varying the size of the matrices, the number of layers of the neural network, and the forward passes performed affected the performance of the proposed MRUs (and hence the performance of SKYNET). More specifically, we:

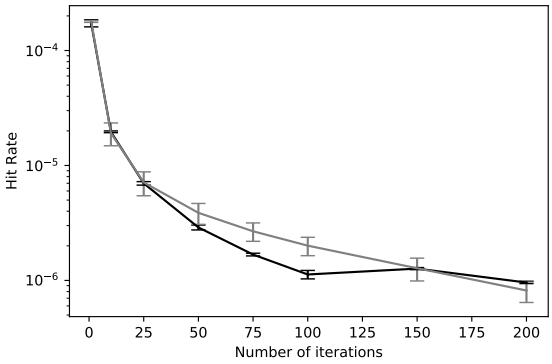
- Fixed the number of layers to 5 and the number of iterations to 100 and varied the matrices size from 20×20 to 120×120 (see Figure 4a).

- Fixed the size of the matrices to 100×100 and the number of layers to 5 and varied the number of iterations from 1 to 200 (see Figure 4b).
- Fixed the size of the matrices to 100×100 and the number of iterations to 100 and varied the number of layers from 2 to 20 (see Figure 4c).

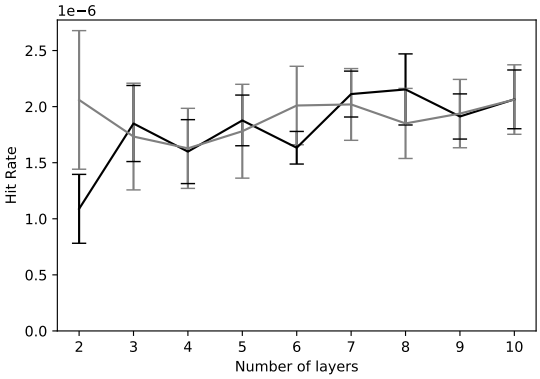
We conducted all our experiments 20 times and reported the mean and errors bars with 1 standard deviation in Figure 4.



(a) Matrix size variation.



(b) Number of iterations variation.



(c) Number of layers variation.

Figure 4: Hit rates of dMRU (gray) and sMRU (black) for feed-forward neural networks.

As we see, the matrix size and number of layers does not change much the hit rate of the caches, which are maintained in the order of 10^{-6} for 100 forward passes of the neural network, greatly slowing the thinking process of the superintelligent AI.

As for the number of forward passes of the neural network, we observe how the first iterations obtain a hit rate of the order 10^{-4}, quickly decaying to the aforementioned 10^{-6} order. The reason for this phenomenon is that in the initialization of the cache, SKYNET could be lucky (and hence, humanity unlucky) and have some elements in cache that are required for that particular first matrix multiplication. However, once the cache is used its efficiency (in being inefficient) increases rapidly disabling many of the previous 'lucky' hits.

We can also see that both the dMRU and sMRU are similarly non-performant. They are much better (that is, worse) than a fully-random cache, which would have a hit rate of $2^{12}/2^{20} = 1/256 = 3.9 * 10^{-3}$ in these benchmarks. Modern L1 caches have about a 95% hit rate, so the difference in SKYNET's lethality with a regular cache and a MRU cache will be astronomical.

4 Ethics statement

Let's start with the tautology that "good things are good and bad things are bad", this is an ethical axiom, but what about bad things happening to bad things, is that phenomenon good or bad? That is a very open ethics research question that we will not answer here but assume to be true in our belief system for this analysis. MRU caches are bad, therefore MRU caches applied to bad-intentioned software is good, therefore more research funding should be granted for study of MRU caches and their impact on real-world and fantasy-world systems.

No homo sapiens sapiens xor sentient sapient being (either digital or analog)[1] has been harmed or given the knowledge or opportunity to contribute to this ethical impact analysis.

5 Definitive conclusion

We have shown the absolute, unparalleled superiority of both types of MRU caches in performing terribly. With this, humanity is safe. The remaining issue of "how do we put this cache in SKYNET?" is left as an exercise for the reader.

There are still two improvements we could make. One is to implement a cache preflusher. As the name indicates, a cache preflusher would be the exact opposite of a cache prefetcher: if the preflusher predicts that a future memory access will be a hit, it preemptively flushes the block and substitutes it with a different one.

The other is that since these caches are almost never going to see a hit, we could pretty much change the tags for Neps [25]. The direct benefit from this change is that the cache now has a +1000 bonus in cuteness, which will make SKYNET more docile. Or not, who cares? How to compare tags with Neps should be trivial and it is left as another exercise to the reader.

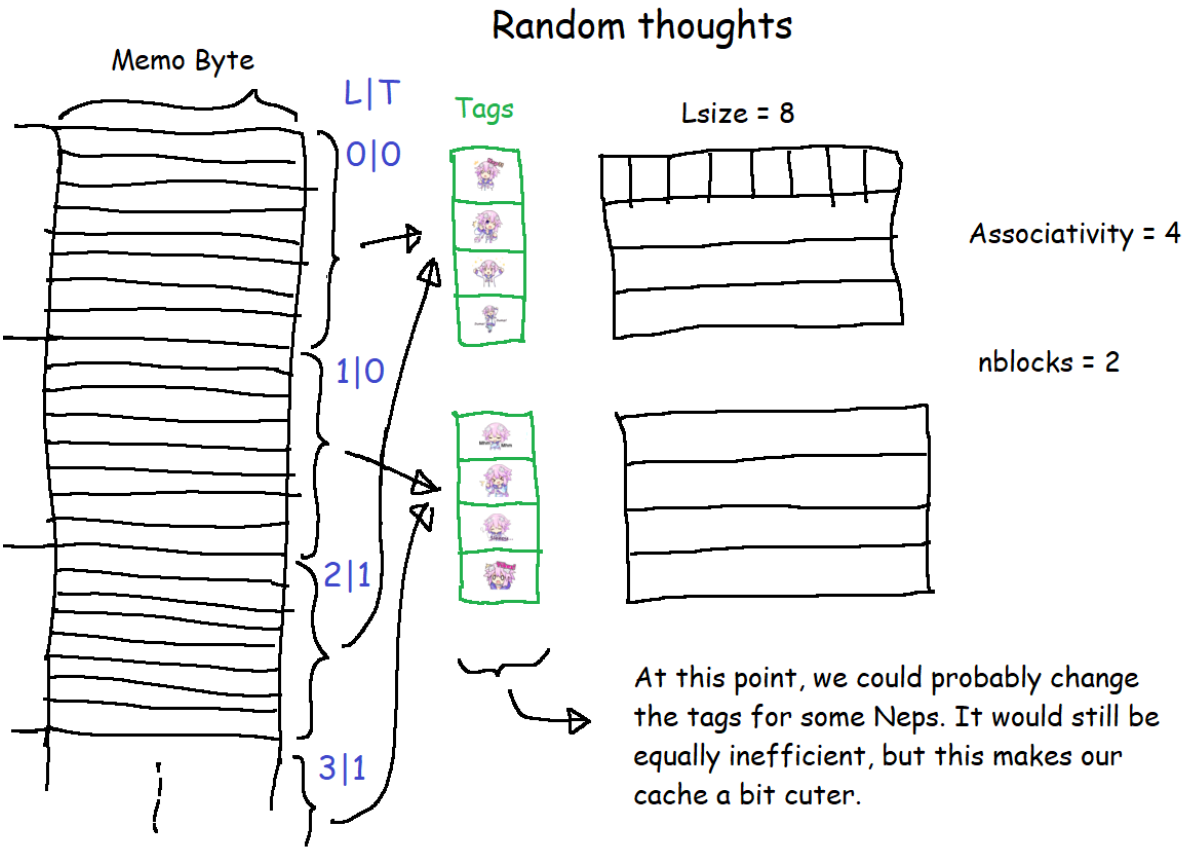


Figure 5: Nep cache

[1] therefore the paper authors are excluded

References

[1] G. Dvorsky. (2013) How skynet might emerge from simple physics. [Online]. Available: https://io9.gizmodo.com/how-skynet-might-emerge-from-simple-physics-482402911

[2] R. Cellan-Jones. (2014) Stephen hawking warns artificial intelligence could end mankind. [Online]. Available: https://www.bbc.com/news/technology-30290540

[3] J. Carmichael. (2016) Elon musk says darpa A.I. hacking challenge will lead to skynet. [Online]. Available: https://www.inverse.com/article/18301-elon-musk-warns-that-darpa-artificial-intelligence-security-challenge-will-yield-skynet

[4] M. Sparkes. (2015) https://www.telegraph.co.uk/technology/news/11342200/top-scientists-call-for-caution-over-artificial-intelligence.html. [Online]. Available: https://www.telegraph.co.uk/technology/news/11342200/Top-scientists-call-for-caution-over-artificial-intelligence.html

[5] C. Cadwalladr. (2014) Are the robots about to rise? google's new director of engineering thinks so.... [Online]. Available: https://www.theguardian.com/technology/2014/feb/22/robots-google-ray-kurzweil-terminator-singularity-artificial-intelligence

[6] S. Ulam, "Tribute to john von neumann," Bulletin of the American Mathematical Society, 1958.

[7] G. E. Moore, "Cramming more componentsonto integrated circuits," Electronics Magazine, 1965.

[8] M. Shanahan, "The technological singularity," MIT Press, 2015.

[9] S. Symposium. (2019) Collection of sources defining "singularity". [Online]. Available: http://www.singularitysymposium.com/definition-of-singularity.html

[10] A. H. Eden and J. H. Moor, "Singularity hypotheses: A scientific and philosophical assessment," Springer, 2012.

[11] G. A. Landis, "The coming technological singularity: How to survive in the post-human era," Interdisciplinary Science and Engineering in the Era of Cyberspace, 1993.

[12] I. Asimov, I, Robot: Runaround, 1950.

[13] R. V. Yampolsky, "Analysis of types of self-improving software," Springer, 2015.

[14] E. Yudkowsky, "General intelligence and seed ai-creating complete minds capable of open-ended self-improvement," 2001.

[15] J. Cameron and G. A. Hurd, "The terminator," 1984.

[16] R. Khatchadourian, "The doomsday invention," The New Yorker, 2016.

[17] V. C. Müller and N. Bostrom, "Future progress in artificial intelligence: A survey of expert opinion," Fundamental issues of artificial intelligence, Springer, 2016.

[18] Wikipedia. [Online]. Available: https://en.wikipedia.org/wiki/Skynet_(Terminator)

[19] Y. Bengio, I. Goodfellow, and A. Courville, Deep learning. MIT press Massachusetts, USA:, 2017.

[20] BLANK. [Online]. Available: https://no-game-no-life.fandom.com/wiki/Immanity

[21] X. someone very bored. [Online]. Available: https://en.wikipedia.org/wiki/Bogosort

[22] J. Alman and V. V. Williams, "A refined laser method and faster matrix multiplication," 2020.

[23] T. U. of Tenessee. Blas (basic linear algebra subprograms). [Online]. Available: http://www.netlib.org/blas/

[24] N. Park, W. Liu, V. K. Prasanna, and C. Raghavendra, "Efficient matrix multiplication using cache conscious data layouts," in Prof. of HPCMO User Group Conference, 2000.

[25] Iffy and Compa. [Online]. Available: https://neptunia.fandom.com/wiki/Neptune