

Technická realizace a implementace prostředků umělé inteligence.

Hardware implemetation of artificial intelligence elements.

Pavel Vyhlídal



Univerzita Tomáše Bati ve Zlíně

Fakulta aplikované informatiky

akademický rok: 2010/2011

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: Pavel VYHLÍDAL

Osobní číslo: A08179

Studijní program: B 3902 Inženýrská informatika

Studijní obor: Informační a řídicí technologie

Téma práce: Technická realizace a implementace prostředků
umělé inteligence

Zásady pro vypracování:

1. Vypracujte literární řešení na téma problematika umělé inteligence, neuronových sítí, evolučních algoritmů a jejich řešení.
2. Prostudujte možnosti jejich realizace pomocí CPU, GPU, GA či analogových obvodů. Porovnejte tato řešení s přírodními ekvivalenty.
3. Navrhněte techniku realizace či simulace NN na těchto prostředcích. Identifikujte části kritické na výkon.
4. Zvolte vhodný problém, na kterém budou patrné rozdíly v technické implementaci. Tyto techniky realizujte či simulujte na dostupných prostředcích v programovacím jazyce C (C++).
5. Diskutujte možný vývoj implementace umělé inteligence, neuronových sítí a evolučních algoritmů se vzrůstajícím výkonem technických prostředků.

Rozsah bakalářské práce:

Rozsah příloh:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

1. ROJAS, R. Neural Networks, Springer-Verlag, Berlin, 1996, ISBN 3-540-60505-3.
2. ZELINKA, I. Umělá inteligence – hrozba nebo naděje. BEN – technická literatura, 2003, ISBN 80-7300-068-7.
3. ZELINKA I., OPLATKOVÁ Z., ŠEDA M., OŠMERA P., VČELAŘ F., Evoluční výpočetní techniky – principy a aplikace, BEN, Praha, 2008, ISBN 80-7300-218-3.
4. GURNEY, Kevin. An Introduction to Neural Networks. 1st edition. CRC Press, 1997. 234 s. ISBN 978-1857285031.
5. ZELINKA, Ivan. Aplikovaná Informatika. Zlín. UTB, 1999. 183 s. ISBN 80-214-1423-5.
6. ZELINKA, Ivan. Umělá inteligence v problémech globální optimalizace. BEN, 2002, 190 s. ISBN 80-7300-069-5.

Vedoucí bakalářské práce:

Ing. Roman Šenkeřík, Ph.D.

Ústav informatiky a umělé inteligence

Datum zadání bakalářské práce:

25. února 2011

Termín odevzdání bakalářské práce:

7. června 2011

Ve Zlíně dne 25. února 2011

prof. Ing. Vladimír Vašek, CSc.
děkan



prof. Ing. Vladimír Vašek, CSc.
ředitel ústavu

ABSTRAKT

Cílem této práce je zhodnocení vývoje technických prostředků pro obor neuronových sítí, evolučních algoritmů jako hardwarového základu pro aplikace umělé inteligence a pokus o odhad jejich dalšího vývoje a jeho dopadů. Práce se skládá ze dvou částí. V teoretické části je popsán současný stav hardware pro realizaci neuronových sítí, v části praktické je provedeno výkonnostní porovnání různých hardwarových řešení do budoucnosti. Součástí je návrh hardware pro neuronovou síť.

Klíčová slova: Neuronové sítě, evoluční algoritmy, umělá inteligence.

ABSTRACT

The aim of this study is to assess developments in neural networks hardware, evolutionary algorithms that are hardware based on artificial intelligence application and attempt to estimate their future development and its impacts. This study consists of two parts. In the theoretical part describes the current state of hardware implementation of neural networks, in the practical part includes a performance comparison of different future hardware solutions. Study contains a draft design for hardware neural network.

Keywords: Neuronal networks, Evolutionary algorithm, Artificial intelligence.

Základním pojmem poznání je omyl, zpočátku samozřejmě přesný. Jak se naše poznání prohlubuje, dostáváme se do druhé fáze poznávacího procesu, v němž omyl vyvracíme. Takže na konci poznávacího procesu nevíme sice nic, ale zato to víme správně.

Cimrmanova teorie poznání

Prohlašuji, že

- beru na vědomí, že odevzdáním bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – bakalářskou práci nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- beru na vědomí, že pokud bylo k vypracování bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze bakalářské práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně

.....
podpis diplomanta

OBSAH

ÚVOD.....	9
I TEORETICKÁ ČÁST.....	11
1 VÝVOJ PROSTŘEKŮ UMĚLÉ INTELIGENCE	12
1.1 NEURON	13
1.2 NEURONOVÁ SÍŤ - MOZEK.....	14
1.3 UMĚLÝ NEURON - PERCEPTRON.....	15
1.4 PERCEPTRONOVÁ (NEURONOVÁ) SÍŤ	16
1.5 ALGORITMY UČENÍ.....	18
2 HARDWARE PRO NEURONOVÉ SÍTĚ	20
2.1 TYPY PROSTŘEDKŮ	20
2.1.1 ANALOGOVÉ OBVODY / NEUROMORFNÍ SÍTĚ.....	20
2.1.2 POČÍTAČE, CPU	21
2.1.3 MNOHOJÁDROVÉ PROCESSORY, (GP)GPU - CUDA	22
2.1.4 HRADLOVÁ POLE (GA), ZÁKAZNÍCKÉ OBVODY (ASIC)	23
2.2 FYZICKÉ PRVKY	24
2.2.1 Transistory.....	25
2.2.2 Memristor	25
2.2.3 Molecular crossbar latch	27
2.3 PAMĚTI.....	27
2.3.1 SRAM	27
2.3.2 DRAM.....	28
2.3.3 Flash EEPROM.....	29
2.3.4 Optická maska, hologram.....	30
2.4 PASIVNÍ PRVKY	30
2.4.1 Sběrnice.....	30
Paralelní sběrnice	31
Kompenzované paralelní sběrnice.....	31
LVDS / 8b10b kódování.....	32
2.4.2 Napájení	32
2.4.3 Chlazení.....	32
II PRAKTICKÁ ČÁST	34
3 TECHNIKY SIMULACE NEURONOVÉ SÍTĚ.....	35
3.1 JEDNOVRSTVÁ NEURONOVÁ SÍŤ BEZ REKURENCE	35
3.2 JEDNOVRSTVÁ NEURONOVÁ SÍŤ S REKURENCÍ	36
3.3 VÍCEVRSTVÁ NEURONOVÁ SÍŤ	36
3.4 OPTIMALIZACE	36
3.5 NÁHRADA REKURENCE.....	36
4 SIMULACE NEURONOVÉ SÍTĚ POMOCI CPU A GPU.....	37
4.1 CPU	37
4.2 GPU-CUDA	38
5 VYUŽITÍ NEURONOVÝCH SÍTÍ.....	42

5.1	AI ZIMA (AI WINTER)	42
5.2	INTEGRACE AI DO SYSTÉMŮ	42
5.3	IMPLEMENTACE NEURONOVÝCH SÍTÍ DO HARDWARE	43
5.4	ROZVOJ AI.....	45
ZÁVĚR		47
ZÁVĚR V ANGLIČTINĚ.....		48
SEZNAM POUŽITÉ LITERATURY.....		49
SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK		51
SEZNAM OBRÁZKŮ		52
SEZNAM TABULEK.....		53
SEZNAM PŘÍLOH.....		54

ÚVOD

Funkce a princip činnosti mozku bylo dlouhou dobu neznámou. Jeho poznávání probíhal jako u ostatních medicínských oborů formou pokus – omyl a na základě pozorování funkčních změn u orgánových onemocnění a poranění. Tento způsob samozřejmě funguje i dnes. Intelkt se přisuzoval nehmotné duši nebo srdci. Hippokrates již situoval myšlení do mozku a k dalšímu rozmachu došlo v období renesace spolu s obecnou anatomíí. Teoretické základy k fungování neuronových sítí se začaly objevovat s objevem neuronu. S neuronovou teorií přišel Santiago Ramón y Cajal koncem 19. století kdy identifikoval oddělené buňky v mozku. Z filozofického hlediska nelze opomenout Čapkův R.U.R. K významnému pokroku v oboru umělých neuronových sítí a umělé inteligence došlo ve 40. letech 19. století významnými objevy především Allana Turinga a John von Neumanna. V té době se také předpokládal prudký rozvoj umělé inteligence a sci-fi zaplnili antropomorfní roboti s jistou dávkou společensky problematické inteligence. Velmi rychle se ukázalo že „umělý mozek“ nebude zkonstruován do několika let, a neuronové sítě a evoluční techniky nenašli uplatnění v několika následujících desetiletích. Přístup k umělé inteligenci se rozdělil na symbolický – přístup shora dolů, pokus o napodobení inteligence prostřednictvím analýzy poznávacích schopností a na konekcionistický – přístup zdola nahoru, pokus o napodobení funkce neuronů a přístupy k jejich učení [1]. Ke znovuobjevení došlo až s příchodem mikroelektroniky – technologií výroby integrovaných obvodů s vysokou hustotou integrace. Ta umožnila implementaci nejprve expertních systémů a fuzzy logiky pro řízení jednoduchých systémů, které nejsou tak náročné na výkon a poté neuronových sítí včetně jejich učení, které jsou již náročnější především na paralelní výkon. Tyto aplikace se již objevují například v rozpoznávání obličejů, burzovní odhady, CAD modelování, řešení logistických problémů a dalších. Zdá se, že jejich příchod bude mít společenský vliv podobný, jako měla industrializace a pásová výroba koncem 19. století. Nedávné vítězství počítače Watson od IBM v oblasti pochopení přirozeného lidského jazyka a kontextu, která byla přisuzována výlučně lidem, ukazuje na zrychlující se vývoj umělé inteligence jež by, podle některých pramenů, mohla vést v technologickou singularitu.

Cílem této práce je zhodnocení vývoje technických prostředků pro obor neuronových sítí, evolučních algoritmů jako hardwarového základu pro aplikace umělé inteligence a pokus o odhad jejich dalšího vývoje a jeho dopadů. Skládá ze dvou částí. V teoretické části je v první kapitole popsán vývoj prostředků pro umělou inteligenci od přírodních ekvivalentů

po současnost a nastíněny způsoby jejich učení. V druhé kapitole jsou vypsány typy hardwarových prostředků s ohledem na jejich vztah k neuronovým sítím. V praktické části jsou ve třetí kapitole popsány způsoby simulace neuronových sítí, ve čtvrté je praktická ukázka algoritmu na CPU a na GPU a porovnána jejich výkonnost. V poslední kapitole je popsáno využití neuronových sítí v praxi a odhad vývoje specifického hardware pro jejich další implementace.

I. TEORETICKÁ ČÁST

1 VÝVOJ PROSTŘEKŮ UMĚLÉ INTELIGENCE

Intelligence. Slovo pochází z lat. „inter-legere“, rozlišovat, poznávat, chápat. Můžeme si pod tímto pojmem představit rozumovou schopnost řešit nově vzniklé nebo obtížné situace; učit se ze zkušeností; adaptovat se na nové podmínky; schopnost pochopení souvislostí a vztahů nebo schopnost abstraktního myšlení.

Umělou inteligencí poté uvažujeme projev, který není produktem lidské mysli. Nedá se určit, která definice by byla nejvíce obecně akceptována, proto uvádím například:

Umělá intelligence je věda o vytváření strojů nebo systémů, které budou při řešení určitého úkolu užívat takového postupu, který - kdyby ho dělal člověk - bychom považovali za projev jeho intelligence. (Minsky, 1967)

Umělá intelligence se zabývá tím, jak počítačově řešit úlohy, které dnes zatím zvládají lidé lépe. (Rich, Knight, 1991)

Některé definice jsou vztaženy k lidskému myšlení, kdy je mozek používán jako etalon intelligence. Nedá se však obecně tvrdit, že by chování podobné či nerozeznatelné od lidského bylo nutně produktem intelligence. Příkladem je argument čínského pokoje, kdy je možné smysluplně odpovídat na položené otázky bez jejich porozumění. Dalším příkladem by byl program ELIZA, který vytvořil Joseph Weizenbaum v polovině 60. let a který je základem propracovanějších chatterbotů a programů pro komunikaci přirozeným jazykem.

Jak již bylo uvedeno v úvodu, přístup k umělé inteligenci můžeme rozdělit na symbolický – přístup shora dolů, snahu o napodobení pomoci analytických a heuristických přístupů a na konekcionistický – přístup zdola nahoru, pokus o napodobení funkce mozku pomoci neuronové sítě [1]. Právě tento přístup je předmětem této práce.

Základem činnosti mozku je neuronová síť. Počet neuronů v této síti je přibližně 10^{11} a jsou propojeny přibližně 10^{15} spoji – synapsemi. Konekcionistický přístup k umělé inteligenci bude tedy pokus o napodobení činnosti neuronové sítě.

Realizaci této sítě můžeme rozdělit na fyzickou a softwarovou umělou neuronovou síť. Ve fyzické realizaci budou živé neurony nahrazeny jejich elektronickou podobou v jejich počtu a množstvím propojů. V softwarové podobě bude tato síť realizována jako numerický model v logických obvodech.

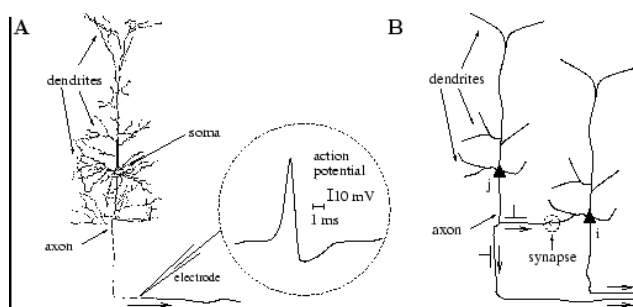
1.1 NEURON

Neuron je komunikační buňka vyskytující se v rámci nervového systému ve velkém množství subtypů, různě vzájemně propojených. Na neuronu jsou patrné oblasti s různou specializovanou funkcí [7]. Na typickém neuronu pozorovat tři hlavní oblasti:

- tělo neuronu, obsahující jádro a hlavní cytoplasmatické organely;
- různý počet dendritů, které odstupují z těla, liší se velikostí, tvarem, přítomností dendritických trnů a větví se různě daleko a hustě do šedé hmoty; a
- jediný axon, který u většiny axonů odstupuje z těla do daleko větší vzdálenosti než kterákoliv z větví dendritického stromu.

Velikost těla neuronu se pohybuje od asi 5 do 100 mikrometrů. Neurony lze rozdělit podle počtu z těla odstupujících výběžků na unipolární s jedním výběžkem, ze kterého se pak rozděluje dendritický strom a axon, bipolární s jedním dendritickým a jedním axonálním výběžkem na opačných stranách či multipolární s mnoho výběžky. Neurony mají různé tvary, lišící se počtem větví dendritických stromů a jejich délkou.

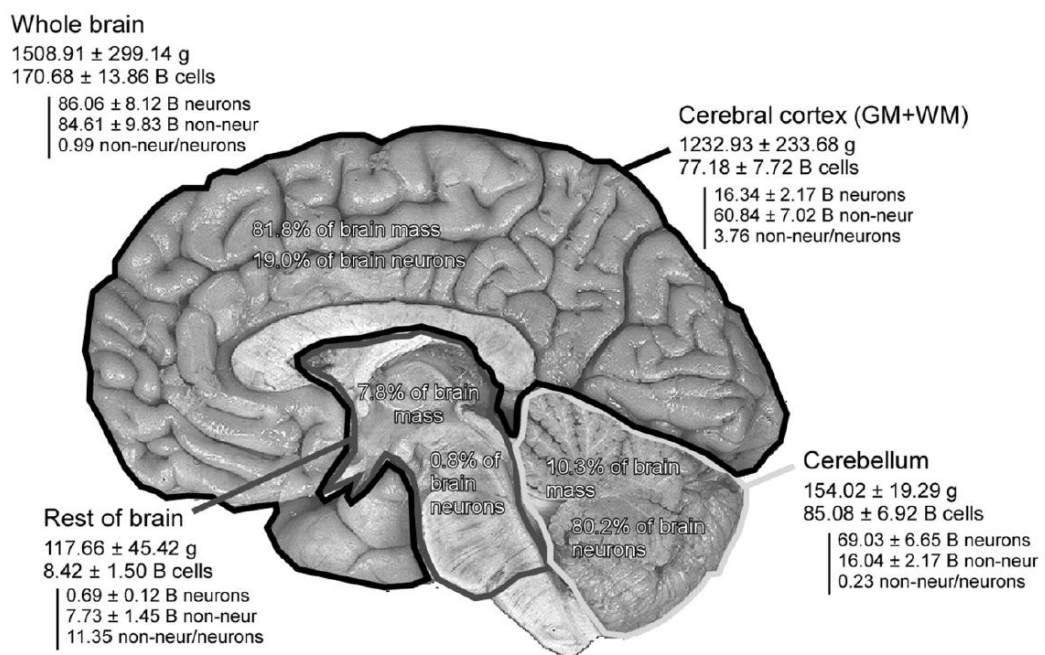
Funkcí neuronu je přenos vzruchů: na vstupní impuls na některém z dendritických výběžků jenž přesáhne prahovou úroveň neuron reaguje přenosem vzruchu do axonu. To proběhne depolarizací buněčné membrány. K excitaci neuronu dochází dle pravidla všechno nebo nic. Pokud excitační potenciál nedosáhne prahové úrovně, neuron nereaguje. Po překročení prahové úrovně se depolarizační vlna šíří všemi směry. Na větší stimulaci pak neodpovídá větší excitací, ale zvýšenou frekvencí generovaných excitačních impulsů. Neurony navzájem komunikují na synapsích pomocí neurotransmiterů. Ty mohou mít charakter inhibiční, excitační nebo modulační. Příklad neuronu s jednoduchým [16] spojením (Obr. 1).



Obr. 1. Přenos impulsu neuronem.

1.2 NEURONOVÁ SÍŤ - MOZEK

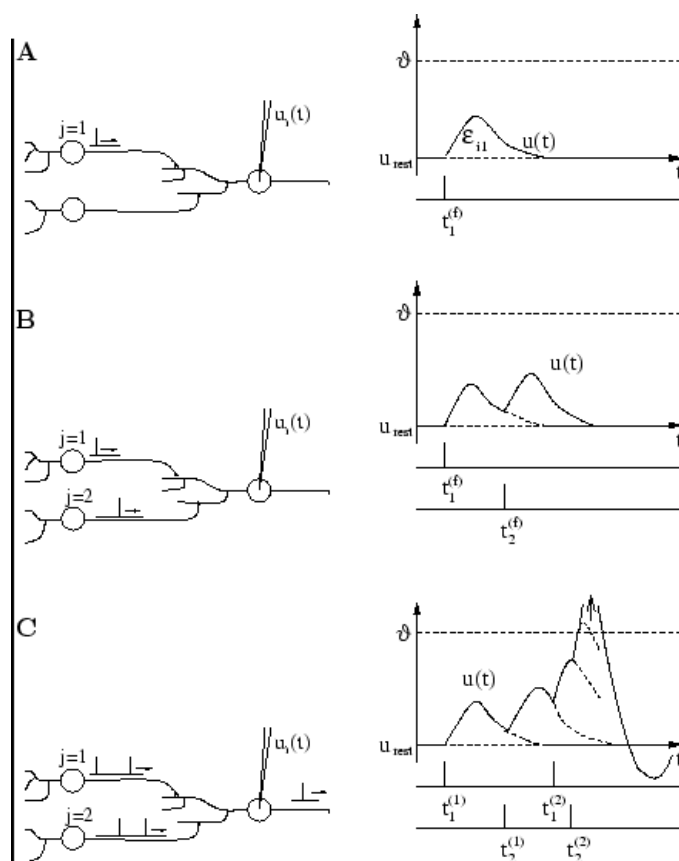
Mozek je funkční spojení velkého množství neuronů, u člověka asi se udává 80 - 100 miliard. Každý neuron má v průměru 7000 synaptických spojení s dalšími neurony, pro Purkyňovy buňky v mozečku se však udává více než 100000 spojení. Počet buněk v jednotlivých částech mozku je uveden například v práci Azevedo et al. (2009), (Obr. 2).



Obr. 2. Počet neuronů v jednotlivých částech mozku, Azevedo et al. (2009).

Z tohoto výčtu vychází poměrně zajímavé zjištění, že většina neuronů se nenachází v koncovém mozku (cerebral cortex - cca. 20%) ale v mozečku (cerebellum - cca. 80%). Mozeček je přitom centrum především koordinace motoriky, volní funkce (včetně pohybu) a intelekt je přisuzován koncovému mozku.

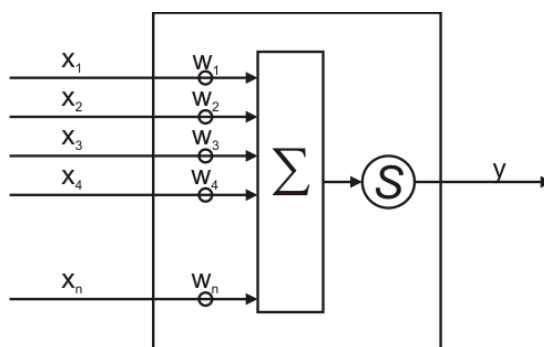
Propojením organických neuronů získáme síť, která má nejen statické, ale i dynamické vlastnosti [16]. Na nich se budou podílet nejen množství v sérii zapojených neuronů, ale i délky drah, jejich průměr jenž má vlastnost na rychlost šíření vzruchu a předchozí stavy neuronů. Na následujícím obrázku (Obr. 3) je příklad dynamických vlastností propojení 3 neuronů.



Obr. 3. Přenosové funkce spojení více neuronů.

1.3 UMĚLÝ NEURON - PERCEPTRON

Perceptron je elektronickou podobou neuronu. Na základě pozorování byl vytvořen jeho matematický model [6], schematicky na obrázku (Obr. 4). Má několik vstupů a jeden výstup. Každý vstup je násoben individuální vahou, takto zpracované vstupy jsou sečteny a zpracovány přenosovou funkcí.



Obr. 4. Perceptron.

$$y = S \sum_{i=0}^n x_i w_i - \Theta$$

kde

y je výstup,

x_i jsou vstupy v počtu n ,

w_i jsou váhy, přináležejícím každému vstupu,

S je přenosová funkce, může být použita skoková změna, sigmoida, hyperbolická tangenta nebo podobná nelineární funkce.

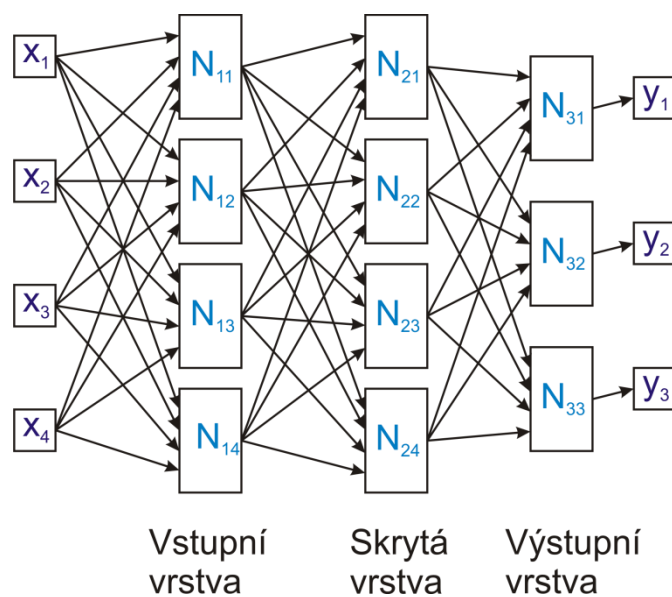
Θ je práh, po jehož překročení dojde k aktivaci.

Toto schéma však vyjadřuje velmi zjednodušenou statickou funkci, funkce přírodního neuronu je mnohem složitější obsahující spoustu dynamických a nelineárních vlastností [17].

Spojené preceptrony tvoří neuronovou síť, která může tvořit vrstvy a může být rekurentně pospojovaná.

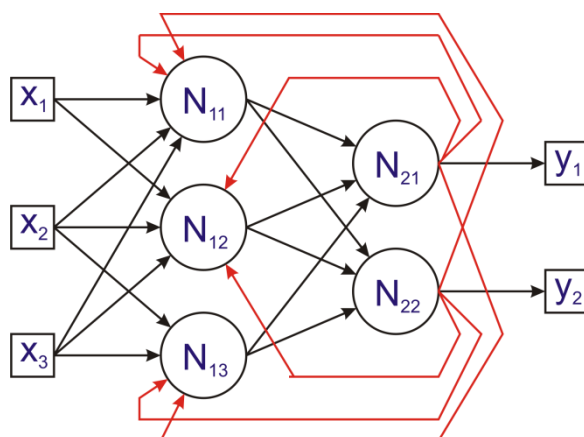
1.4 PERCEPTRONOVÁ (NEURONOVÁ) SÍŤ

Zde je příklad třívrstvé sítě (Obr. 5). Má čtyři vstupy, tři vrstvy, prostřední vrstva je skrytá. Každý neuron má informace ze všech předchozích výstupů. Výstupy neuronů můžeme považovat za abstrakci vstupů. Jednotlivé výstupy vrstev neuronů tak můžeme považovat za vyšší a vyšší úrovně abstrakce vstupních dat.

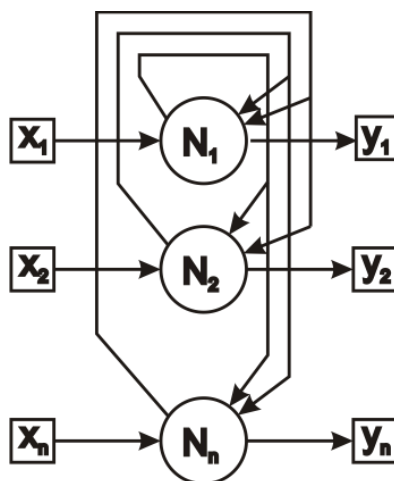


Obr. 5. Třívrstvá neuronová síť s jednou skrytou vrstvou.

Kromě vstupů x_i mohou být na vstupy neuronů přivedeny i výstupy y_i z neuronů stejné nebo vyšších vrstev jak je zobrazeno na následujících obrázcích (Obr. 6), (Obr. 7).



Obr. 6.: Neuronová síť s rekurencí.

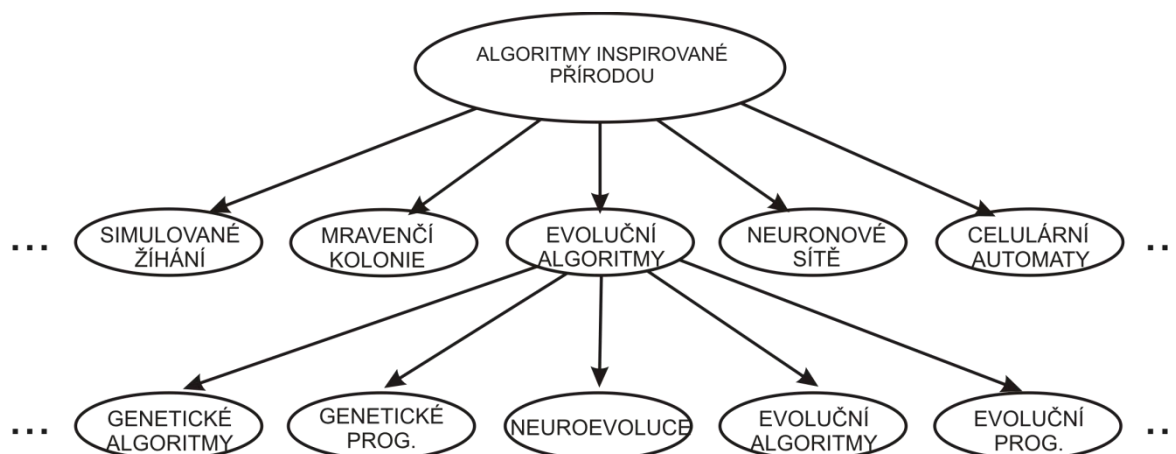


Obr. 7: Hopfieldova síť.

1.5 ALGORITMY UČENÍ

Samotné neuronové sítě nic nevyřeší, bez možnosti jejich učení. Zde bych sítě rozdělil na již naučené bez dalšího učení, jednorázové učení pro řešení problému a na trvale se učící, nějakou formou synaptické plasticity. Například u fotoaparátu očekáváte, že bude schopen detekovat scény, rozpoznávat obličeje již od výrobce, aniž by jste to neuronovou sítí museli učit. Výrobce naopak bude muset mít prostředky, které na základě množství vzorů očekávání vytvoří vhodnou matici vah, kterou pak do Vašeho přístroje nahrají. U jiných sítí, řešících trvale se měnící podmínky, například burzovní úlohy, sociální sítě budete očekávat trvalou plasticitu, schopnost trvalého učení, adaptace.

Tyto různé úlohy budou mít také různé požadavky na hardwarové řešení. Příklad rozdělení algoritmů inspirovaných přírodou je na následujícím diagramu (Obr. 8).



Obr. 8. Algoritmy inspirované přírodou.

U naučených systémů to bude nízká spotřeba, cena a opakovatelnost. Tomu odpovídá simulace sítě na CPU, ev. s hardwarovou akcelerací, například pomocí DSP. Takto jsou řešeny v současné době fotoaparáty, systémy rozpoznávání znaků.

Pro učení těchto sítí bude potřeba systém s vysokou synaptickou a neuro plasticitou (schopností konfigurovat strukturu sítě a její váhy) a schopný simulovat celé populace těchto sítí pro evoluční techniky výběru optimálních řešení. Tomu odpovídají výkonnější servery ev. jejich clustery, využívající akcelerace výpočtů např. pomocí GPGPU.

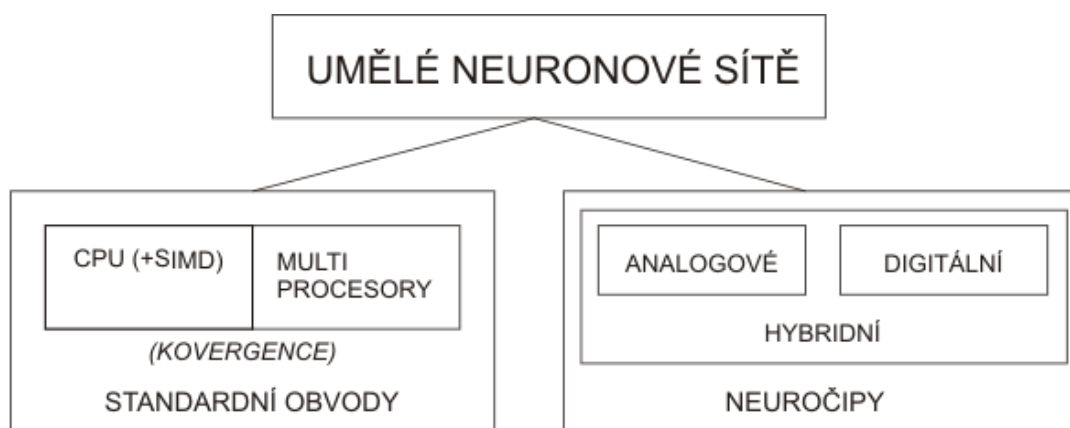
Adaptivním systémům je nejbližší lidské myšlení, tedy funkce mozku. Jeho vlastností je schopnost samoorganizace, strukturální adaptace na nové úkoly. Další vlastností je také neopakovatelnost, každý je unikátní a jeho vlastnosti nelze přenést na jiný. Budou-li takovéto systémy realizovány pomocí ryze deterministických prostředků – tedy například pomocí paměti RAM a digitálních procesorů, bude tato neopakovatelnost odstraněna. Naopak budou-li realizovány na prostředcích s vysokou mírou neurčitosti, jako jsou například analogové neuromorfí sítě, bude tato neopakovatelnost zachována. Tento rozdíl je významný pro evoluční hledání optimálního řešení. Nebudeme-li schopni takovou stochastickou síť rozdělit do unifikovaných abstraktních vrstev s unifikovanými vstupy, nebude jednoduché tyto sítě vyvíjet s použitím genetických algoritmů.

2 HARDWARE PRO NEURONOVÉ SÍTĚ

Výpočetní výkon se obvykle udává v jednotkách MIPS (milionů instrukcí za sekundu) nebo FLOPS (M/G/T/PFLOPS – operací s pohyblivou desetinou tečkou). Moje malá kapesní kamera dokáže na malou baterii nejen přehrávat – dekódovat několik hodin full-HD videa, ale i nahrávat - kódovat, což je úloha ještě náročnější. Dvoujádrový procesor v desktopu na frekvenci 3 GHz se spotřebou 60W nezvládne ani dekódování stejně tak jako miniaturní přehrávač MP3 s baterií velikosti pětikoruny dokáže dekódovat stream mp3 několik desítek hodin, zatím co procesor 486 na 66MHz toho nebyl schopen. MIPS a GFLOPS nejsou vždy směrodatným ukazatelem specifického výkonu a pro neuronové sítě to platí obzvláště. Komunikační systémy CDMA, GPS, WIFI, DVB, které byly před 20. lety velké krabice jsou dnes přívěsky napájené z USB portů. Za touto změnou je vývoj specifického hardware, například DSP, speciálních kóderů či kryptoprocessorů. Simulace mozku se spotřebou asi 20W je zatím nepředstavitelný problém i na supercomputerech se spotřebou několika MW.

2.1 TYPY PROSTŘEDKŮ

Hardwarové prostředky pro umělé neuronové sítě můžeme rozdělit podle jejich produkce na řešení pomocí standardních, v současnosti běžně vyráběných obvodů a na speciální obvody označované jako neuročipy. Ty můžeme, jak je znázorněno na diagramu (Obr. 9) na analogové, digitální ev. jejich hybridní kombinaci.



Obr. 9. Rozdělení umělých neuronových sítí.

2.1.1 ANALOGOVÉ OBVODY / NEUROMORFNÍ SÍTĚ

jsou základem fyzické umělé neuronové sítě. Analogové obvody by byly nejpřirozenějším prostředkem pro simulaci přírodních neuronových sítí. Jejich hlavní výhodou je umístění

paměti a vyhodnocovacích obvodů na jednom místě. Nevzniká tak na rozdíl od von Neumanovy (ale ne jen jeho) architektury počítače úzké hrdlo mezi pamětí (v tomto případě s maticí vah) a procesorem, cyklicky tyto váhy načítající. Dále na rozdíl od diskrétně pracujících CPU (především se jedná o diskretizaci v čase) zpracování probíhá spojitě a odpadá problém se synchronizací vrstev a rekurencí. Přesto se analogové sítě zatím, kromě experimentálních zařízení, příliš nerozšířili. Při současné technologii je počet digitálních i analogových obvodů na ploše integrovaného obvodu podobně limitován, takže počet funkčních „analogových neuronů“ na čip se nebude příliš lišit od počtu „digitálních neuronů“. Současné obvody pracují na frekvenci v řádu GHz a analogovými obvody bychom tak získali sice extrémně rychlou, ale co do počtu neuronů velmi jednoduchou síť. To by bylo účelné pro aplikace, které jsme však schopni řešit deterministicky, například pomocí DSP. Digitální obvod nám umožní multiplexně simulovat mnoho neuronů a jejich váhy a modulární propojení řešit pomocí obsahu paměti. Dále je problém s tak masivním počtem propojů na procesoru. Digitální obvod nám umožní vytvoření úzkých, ale rychlých sběrnic, zatím co analogová síť by byla tvořena také obrovským počtem propojů, což je při současné technologii obtížně realizovatelné. Dalším problémem je rušení a opakovatelnost. Digitální obvody pracující pouze se dvěma úrovněmi (s výjimkou některých flash pamětí) jsou mnohem odolnější vůči rušení a vzájemnými vazbami než analogové obvody. U digitálních obvodů se rozlišují stavy funguje/nefunguje = dobrý/vadný, a všechny „dobré“ poskytují v pracovních mezích totožné výsledky. Analogové obvody mají svoje tolerance a každý by při stejném nastavení a vstupu dal trochu jiný výsledek. To by byl značný problém při reprodukovatelnosti.

2.1.2 POČÍTAČE, CPU

Nejdostupnějším prostředkem k simulování neuronové sítě je počítač s CPU. Výhodou je nízká cena a množství software, které lze pro účely simulace neuronových sítí použít. Nevýhodou však je sériové zpracování dat. Do paměti RAM lze uložit velké množství vah, dle délky jejich slova (1-32 bit, float) řádově 4-200 miliard, což například při počtu 1000 vstupů jednoho neuronu znamená 4-200 milionů simulovaných neuronů. Problémem však zůstane opakovací frekvence, která bude závislá na rychlosti CPU a na přenosové rychlosti sběrnice paměti. Rychlost zpracování v CPU můžou výrazně urychlit větší počet jader a operace SIMD, označované jako MMX, SSE(x), které umožní provedení většího množství operací v jednom taktu (SIMD – single input, multiple data). Slabým místem i tak zůstane úzké hrdlo mezi CPU a pamětí RAM. Ta je připojena k CPU pomocí sběrnice (FSB). Její

rychlost se v současné době pohybuje okolo 15-20 GB/s na kanál, tedy 30-60 GB/s při dual/triple-channel. To by například při použití 8-bit váhy a 1000 spojů na neuron znamenalo simulaci 30-60 milionů neuronů /s. Toto lze zlepšit sklularizací, technikou blokového zpracování, čímž však bude omezená případná rekurentnost sítě.

2.1.3 MNOHOJÁDROVÉ PROCESSORY, (GP)GPU - CUDA

Zpracování grafických dat je výpočetně náročný úkol, který roste s rozlišením a zpracovávanou úlohou. Samotnému CPU by sériové zpracování jednoho obrazu by trvalo neúměrně dlouhou dobu, proto se součástí grafických adaptérů staly grafické koprocesory, které byly schopny vykonávat sice jednoduché, ale masivně paralelní úlohy, například vyplňování ploch texturou, numerické výpočty vertexů. Hlavním motorem tohoto rozvoje byly 3D akční hry které pro renderování scén vyžadovaly grafické procesory s výkonem mnohonásobně převyšujícím CPU. Tento výkon však zůstal kromě her nevyužit a proto se začaly grafické procesory unifikovat – místo proprietárních grafických nástrojů se vytvořili jednotná procesorová jádra, podobná CPU, avšak s menší, uzpůsobenou instrukční sadou. Tyto unifikované koprocesory - GPGPU (General-purpose computing on graphics processing units) například s architekturou CUDA (Compute Unified Device Architecture) se staly základem pro zpracování mnoha dalších úloh než renderování scén. Běžně se využívá například k dekódování komprimovaného videa, kde by CPU nestíhal v reálném čase a tím CPU odlehčí a vzhledem k jednodušší a specializovanější architektuře zvládne totéž s menší spotřebou energie. Další úlohy však narážejí na paralelizovatelnost úloh. Ne všechny úlohy lze rozdělit do vláken, a tak i využití vícejádrových procesorů nebývá vysoké. Simulace neuronových sítí však paralelizovatelná je.

Problémem zůstává úzké hrdlo mezi GPGPU a pamětí, v níž jsou váhy uloženy. GPGPU mívají vlastní paměť na přímo na čipu, podobně jako CPU, paměť na grafické kartě (kromě sdílených řešení) a mají přístup do hlavní RAM. Paměť na kartě bývá omezena na cca. 1-2GB. Její rychlost je vyšší než u hlavní RAM, u GDDR5 přibližně 5 Gbit/s na linku pro typ GDDR5. Počet linek se dle GPGPU pohybuje od 64 až k tisíc, podle výkonnosti karty. Při 512. bitové sběrnici to znamená až 2,5 Tb/s (cca. 300 GB/s) [4]. To je o řád více, než rychlost sběrnice hlavní RAM Slabým místem tedy zůstane přenosová kapacita sběrnice paměti.

2.1.4 HRADLOVÁ POLE (GA), ZÁKAZNÍCKÉ OBVODY (ASIC)

Tyto obvody jsou neoptimálnější, ale také na vývoj nejnáročnějším způsobem řešení. Umožní vytvoření systému šitého na míru s minimální spotřebou, maximálním výkonem a s minimálním počtem součástek. Prakticky každý systém, který na začátku byl řešen větším množstvím univerzálních obvodů, byl po spuštění masové produkce převeden do podoby hradlových polí nebo zákaznických obvodů. CPU, GPU, paměťové řadiče, north/south bridge byly dříve realizovány z univerzálních obvodů než se z nich vytvořili specializované zákaznické obvody. Bude-li tedy požadavek na masovou produkci hardwarových neuronových sítí, budou i ony realizovány pomocí speciálních obvodů.

Na trhu je několik typů obvodů odstupňovaných dle komplexnosti řešení. Obvody PAL a CPLD jsou pole konfigurovatelných buněk, obvody FPGA jsou polem programovatelných hradlových polí s vyšší flexibilitou oproti CPLD. Všechny tyto obvody jsou konfigurovatelné softwarově, to znamená, že struktura obvodu se nahrává po spuštění zařízení z pamětí. ASIC obvody jsou již plně zákaznické obvody, jejich struktura je daná výrobou. Jejich vývoj je nejnáročnější protože výroba masek je cenově náročná a nelze opravit případné chyby softwarovou záplatou jako u PAL – FPGA. I zákaznické obvody mohou být tvořené polem předpřipravených hradlových polí nakonfigurovaných maskou spojů při výrobě nebo mohou tvořit celé zákaznické systémy. Takto jsou vyráběny paměti, procesory.

Tab. 1. Výběr obvodů FPGA firmy Xilinx.

Features	Artix-7	Kintex-7	Virtex-7	Spartan-6	Virtex-6
Logic Cells	352,000	480,000	2,000,000	150,000	760,000
BlockRAM	19Mb	34Mb	85Mb	4.8Mb	38Mb
Total Transc. Bandwidth	211Gb/s	800Gb/s	2,784Gb/s	50Gb/s	536Gb/s
Memory (DDR3)	1,066Mb/s	1,866Mb/s	1,866Mb/s	800Mb/s	1,066Mb/s
I/O Pins	600	500	1,200	576	1,200

V uvedené tabulce (Tab. 1) jsou parametry vybraných obvodů FPGA [15] od firmy Xilinx. Z tabulky vyplývá, že datová propustnost obvodů je vysoká – u nejvyšší verze obvodu Virtex-7 2,7Tb/s (cca. 350 GB/s). K obvodu lze připojit paměti DDR3 do 933MHz což znamená cca. 1,8 Gbit/s na bit, vzhledem k velkému počtu I/O pinů je ale možné podle

podtypu obvodu připojit větší množství 72 bitových DDR3 modulů než k CPU. Podobně firma Altera vyrábí sérii Stratix V [20], s přenosovým pásmem 930 Gbit/s (116 GB/s) a možností připojení až 7 modulů DDR3 na 800MHz, tedy celkem 90 GB/s.

Dále byly a jsou vyráběny i obvody pro určené přímo pro neuronové sítě. Vyvíjeli se hlavně v době, kdy výkon CPU byl ještě nízký pro aplikace NN a GPU se ještě nevyráběli. Jejich použití bylo v systémech rozpoznávání textu. V současné době je jednodušší variantou simulovat umělou neuronovou síť na FPGA nebo použít výpočet na GPU. Zde jsou uvedeny příklady vyráběných neuročipů (Tab. 2).

Tab. 2: Fyzické řešení neuronových sítí.

Název	Architektura	Typ	Učení	Přesnost	Neuronů	Synapsí	Rychlost
Intel ETANN	Feedforward, ML	Analog	Ne	6bx6b	64	10280	2GCPS
Synaptics Silicon Retina	Neuromorphic	Analog	Ne	--	48x48	Odporová síť	--
Micro Devices MD1220	Feedforward, ML	Digital / slice	Ne	1bx16b	8	8	1,9MCPS
Philips Lneuro-2.3	--	Digital / slice	Ne	16-32b	12 PE	--	720MCPS
Inova N64000	GP, SIMD, Int	Digital / SIMD	Ne	1-16b	64 PE	256k	870MCPS
Hecht-Nielson HNC 100-NAP	GP, SIMD, float	Digital / SIMD	Ne	32b	4 PE	512k	250MCPS
IBM ZISC036	Radial Basic Function	Digital / RBF	ROI	8b	36	64x36	40kpat/s
Silicon Recognition ZISC 78	Radial Basic Function	Digital / RBF	KNN, L1, LSUP	--	78	--	1Mpat/s
AT&T ANNA	Feedforward, ML	Hybrid	Ne	2bx6b	16-256	4096	2,1GCPS
Nesa Research Neuralclassifier	Feedforward, ML	Hybrid	BP	--	16	256	3,0GCPS

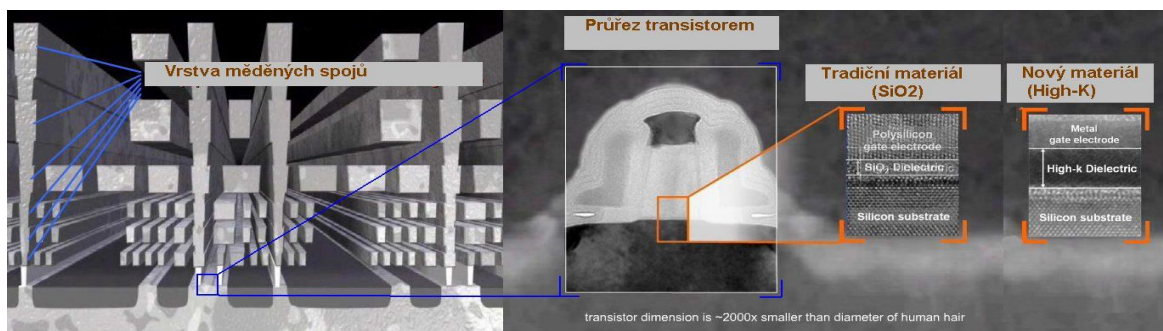
2.2 FYZICKÉ PRVKY

Jsou elementární částí hardware. Právě na nich je závislý nárůst výkonu obvodů. Obecně se dá říci, že vývoj probíhá v souladu Moorovým zákonem [5]: Podle něho se množství prvků (transistorů) na čipu zvýší dvojnásobně každé dva roky. Podobnou závislost však lze najít i u ceny za transistor na čipu, kapacitu pamětí, kapacitu pevných disků, relativní výkon obvodů. Předpokládá se, že tato závislost bude platit i nadále, i když vývoj stávajících technologií kolem roku 2020 narazí na vlnové a kvantové bariéry, po nichž již

nebude možné prosté kvantitativní zmenšování struktur, ale bude nutná i kvalitativní změna fyzikálního principu.

2.2.1 Transistory

Současná technologie řeší tyto prvky jako spínače MOS, tedy obvod s proměnným odporem řízený napětím na řídící elektrodě. Rychlost jejich spínání dosahuje řádu jednotek GHz, ev. v desítkách GHz. Planární technologie výroby MOS obvodů se průběžně vyvíjí cca. od 70. let minulého století a postupně se zvyšuje jejich rychlost a snižují se rozměry a spotřeba. V současné době začíná být vývoj této technologie limitován schopností produkovat struktury s menšími rozměry, kde jsou omezením optické vlastnosti litografické masky, kvantové jevy které se začínají u malých rozměrů výrazně negativně projevovat a při rostoucí hustotě aktivních prvků začíná být problém chlazením – odváděním velkého množství tepla z malé plochy. Řešením je zkracování vlnových délek osvětlení, imerzní litografie a použití vícenásobné masky. Kvantové jevy způsobující proudové svody na hradlech se dočasně vyřešili použitím izolačních materiálů s vysokou permitivitou, jak je vyobrazeno na obrázku (Obr. 10), kde je znázorněno použití High-K materiálu v hradlech transistorů z prezentace technologie firmy Intel [18]. Tato změna umožnila přechod z 65nm na 45nm technologii.



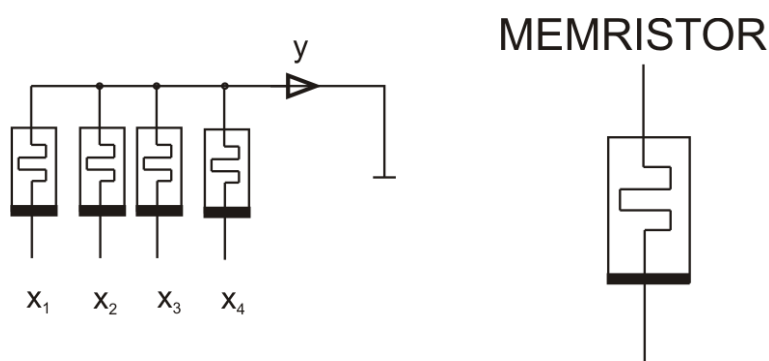
Obr. 10. Ukázky z prezentace využití High-K izolačních vrstev firmy INTEL.

Očekává se, že trend zmenšování struktur bude vyčerpán na technologii 5 – 10 nm, kdy již nebude možné použít stávající metody litografie a problematické již budou kvantové jevy. Východiskem může být principiální změna funkce nebo výrobou vertikálních struktur, tj. přechod do 3D stuktur.

2.2.2 Memristor

Memristor – součástka, jejíž existenci předpověděl v roce 1971 Leon Chua, prakticky realizována až počátkem tohoto století, především firmou HP. Ta do jejich vývoje vkládá

velké naděje. Memristor je čtvrtá z elementárních pasivních součástek - Resistor, Capacitor, L induktor a Memristor. Na rozdíl od předchozích třech má schopnost „pamatovat“ si stav, jenž je výsledkem integrace procházejícího proudu. Její nedávná realizace byla dána technologicky – funkční memristor je konstruován z vrstev silných jen několik nanometrů – vrstev řádově desítek či stovek atomů. To je v současné době technologický problém, nicméně s vývojem výroby nanostruktur může být brzy překonán. Memristory jsou použitelné například v „neuromorfních“ obvodech. Mohly by tak tvořit základ fyzické analogové neuronové sítě. Každý memristor by byl elektronickou náhradou synapse. Vzhledem k jejich velikosti by výsledná síť mohla mít malé rozměry. Memristor navíc v sobě spojuje funkci paměťovou a schopnost učení. V tom se podobá přírodnímu ekvivalentu. To může být i výhodou, protože vhodnou konstrukcí řídicích obvodů lze hodnotu synapse nejenom číst, ale také měnit její hodnotu. Stejnou vlastnost mají také přírodní neuronové sítě, které neobsahují externí elementy nebo obvody, jež by její strukturu či nastavení vah řídili. Taková síť se pak vyznačuje schopností samoorganizace – neuroplasticity, kdy se její struktura a váhy mění dle řešené úlohy. Nevýhodou memristoru je nemožnost externí adjustace hodnoty, která se navíc čtením mění. Také současná tolerance výroby je natolik velká, že síť je těžké replikovat. Velký pokrok byl učiněn v rychlosti, jakou je schopen memristor měnit svoji hodnotu. Příliš pomalá změna parametrů by znamenala pomalou, a energeticky náročnou schopnost adjustace jejich hodnot.

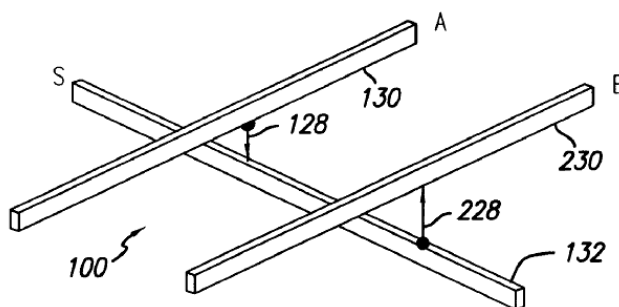


Obr. 11. Memristorová síť a memristor.

Na obrázku (Obr. 11) je schematická značka memristoru. Dále je zobrazen jednoduchý perceptron, kdy se napětí na vstupech $x_1 - x_4$ násobí odporem rezistorů $M1 - M4$ a sčítá. Výsledkem je proud y , představující výstup perceptronu.

2.2.3 Molecular crossbar latch

Jde o patent firmy HP - „Molecular crossbar latch“ [2], (Obr. 12). Jde o matici „molekulárních spínačů“ s vyhodnocovacími obvody, které by mohly nahradit v současné době používané tranzistory v době, kdy současné technologie narazí na kvantové bariéry. Takovéto logické obvody by byly konstrukčně menší a rychlejší.



Obr. 12. Molecular crossbar latch z patentu firmy HP.

2.3 Paměti

Neurony se skládají z těla neuronu a jeho synapsí. Synapse představují váhy a jejich nastavení určuje funkci neuronu. Nastavení vah může být uloženo u analogových sítí přímo ve fyzickém prvku, u simulovaných sítí je váha uložena v digitální paměti.

RAM – Random Access Memory – paměť s náhodným přístupem. Jejím protějškem by byla paměť se sekvenčním přístupem, nicméně při bližším zkoumání se tento rozdíl do jisté míry setře. Podle technologie výroby můžeme rozdělit na DRAM – dynamickou paměť, kde hodnota je uložena v podobě náboje na (postupně vybíjejícím se) kondenzátoru a SRAM, kde je hodnota uložena v klopném obvodu tvořeném 4 nebo 6 tranzistory.

2.3.1 SRAM

Výhodou SRAM je rychlý přístup k obsahu buňky ale nevýhodou velká zastavěná plocha, tedy cena a malá kapacita. Vzhledem k rychlosti je tento typ paměti použit v CPU v registrech a L1 cache paměti, kde pracují na frekvenci procesoru a dále v L2 a ev. L3 (společné pro více jader) cache paměti, kde pracují na rychlosti nižší, ale blízké rychlosti procesoru. Kapacita těchto cache pamětí je důležitá pro výpočetní rychlost – je-li velikost zpracovávaného programu a dat tak nízká, že se vejdou do těchto pamětí, odpovídá rychlost výpočtu přístupové době k nim. Pokud však tuto velikost překročí, začne být

obsah swapován, a data se začínou číst z hlavní paměti RAM která bývá na pomalejší sběrnici a je typu DRAM, což sebou přináší další omezení.

U současných procesorů je velikost těchto pamětí přibližně:

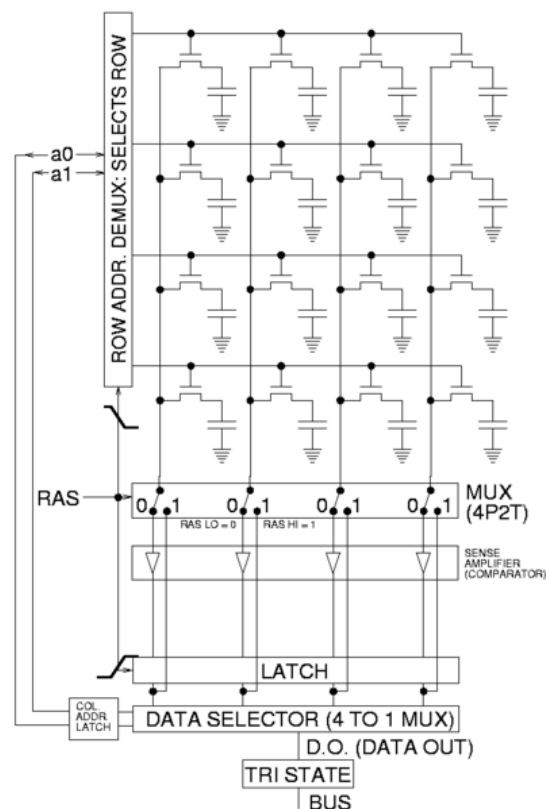
L1 cache – 64+64 kB

L2 cache – 256 kB

L3 cache – 2-4 MB, společná pro všechny jádra na procesoru.

2.3.2 DRAM

Výhodou DRAM je malá zastavěná plocha, tedy velká kapacita čipu, nevýhodou je destruktivní čtení a dlouhá doba přístupu. Informace je uložena jako náboj na kondenzátoru nepatrné kapacity, jehož hodnota je vyčítána po řádcích napětím přivedeným na řídicí elektrodu MOS transistoru přináležejícímu ke každému kondenzátoru v řádku, jak je ukázáno na schématu paměti DRAM 4x4 buňky (Obr. 13).



Obr. 13. Vnitřní struktura paměti DRAM,

Glogger at en.wikipedia.

Tento způsob přístupu umožňuje konstruovat sice miniaturní buňky informace, ale za cenu dlouhého přístupu k jednotlivé informaci. Zde je nevýhoda pamětí DRAM. Zatím co jejich

kapacita v PC se za posledních 20 let zvýšila přibližně 4000x, přenosová rychlost paměťové sběrnice asi 100-200x, doba prvního přístupu k náhodné adrese se snížila cca. 5x. Za vysokými přenosovými rychlostmi stojí zvýšení šířky sběrnice a také velká šířka interní sběrnice sloužící k přednačítání dat po zadání adresy. Například u GDDR5 s šířkou sběrnice 32 bit je při zadání adresy přednačteno 256 bit ze zadané a následujících adres a ty jsou při dalším (sekvenčním) čtení odesílány. Při zadání jiné adresy je však třeba opět čekat na první data cca. 25 ns. Tyto paměti jsou tedy v neuronových sítích vhodné k ukládání vah, které jsou cyklicky sekvenčně čteny, zatímco pro náhodné přístupy je vhodná (nákladnější) paměť SRAM. Také z tohoto důvodu jsou cache paměti (L1 – L3) tvořeny pamětmi SRAM. Jejich kapacita, viz. kapitola SRAM, je nízká, proti přibližně 2-8 GB hlavní DRAM.

Paměti DRAM bývají označovány jako (G)DDR – Double Data Rate, což označuje řízení synchronizačních dat s dvojnásobnou rychlostí dat oproti hodinovému taktu (clock), SDRAM - Synchronous Dynamic Random Access Memory, přičemž slovem synchronní je myšlena závislost časování paměti na vstupu hodinového taktu (clock).

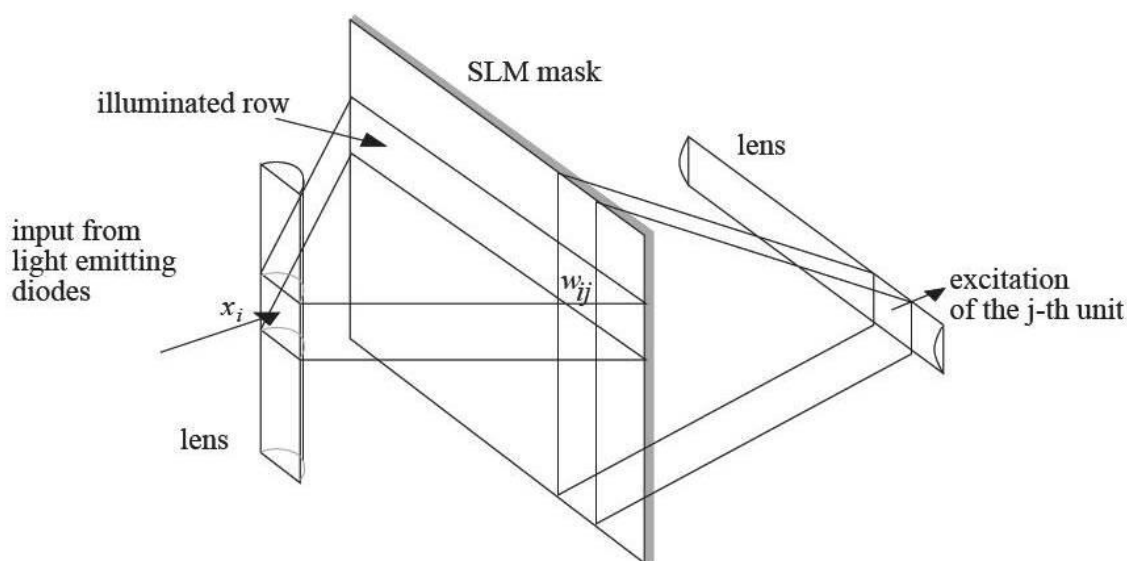
2.3.3 Flash EEPROM

Flash EEPROM je typem nonvolatilní (na napájecím napětí nezávislé) paměti, ve kterých je obsah uložen v kapacitním prvku jehož mazání je řešeno blokově. Na rozdíl od pamětí DRAM nemají destruktivní čtení. Rychlost čtení není vysoká a je podobně jako u pamětí DRAM zvýšena velkou šířkou interní sběrnice. Problémem je změna dat, při změně třeba jen jednoho bitu se musí vymazat vždy celý blok velikosti je řádu kB a celý jej poté zapsat. Při masivně zvýšené šířce interní datové sběrnice můžou tyto paměti poskytovat vysoký datový tok, a tyto paměti tak můžou být vhodné pro ukládání vah, které se často nemění. Výhodou je trvalá přítomnost vah v paměti (nonvolatilní paměť), doba mazání však u těchto pamětí vylučuje multiplexní provoz, kdy by byly váhy cyklicky měněny. Také životnost těchto pamětí bývá omezena na řádově 0,1 – 1 milion zápisů. Tyto paměti se vyrábí ve dvou typech – NOR a NAND, lišící se strukturou buňky (NOR a NAND jen „připomíná“ jejich strukturu), jež se liší rychlostí a kapacitou. Vyšší rychlost NAND má nevýhodu v jejich menší kapacitě. Hodnota buňky v těchto pamětech bývá také uložena částečně analogově, kdy pomocí více úrovní (např. 4) náboje je možné uložit více bitů na jednu buňku. To ale také snižuje jejich rychlost. Tyto paměti se také vyznačují jistou

chybovostí, ne všechny buňky jsou bezchybné, to je poté řešeno blokově a pomocí kontrolních součtů, přístup k paměti pak řídí logika na čipu paměti.

2.3.4 Optická maska, hologram

Optická neuronová síť je jednou z variant fyzického provedení [14]. Jako vstupy se používají zdroje záření, váhy jsou představeny maskou kde nastavení vah je dáno propustností masky. Výstupem je fotodetektor (Obr. 14).



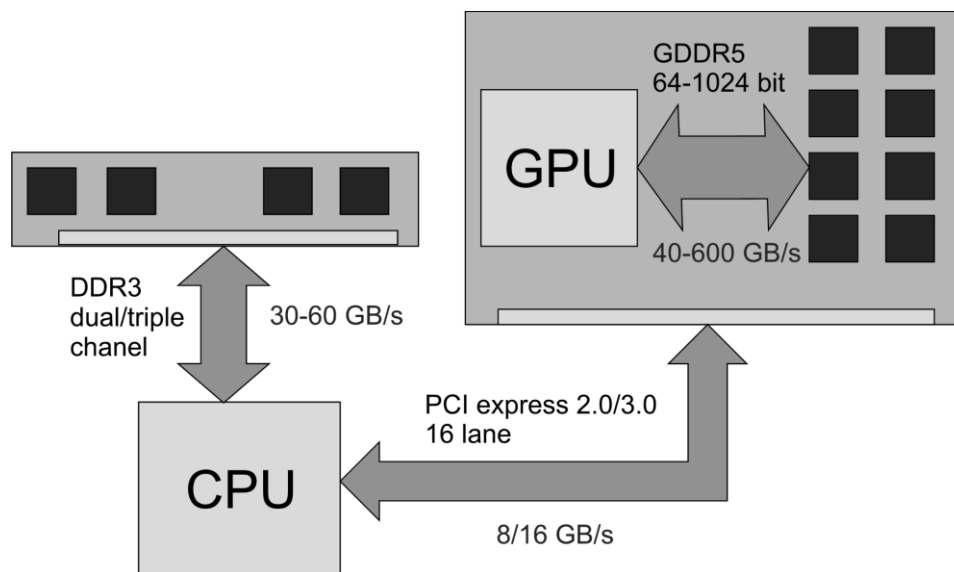
Obr. 14. Schéma optické varianty neuronové sítě.

2.4 Pasivní prvky

Nejen aktivní prvky, jako paměti, procesory a řadiče mají vliv na výkon systému. Důležité jsou také části, jako jsou datové sběrnice pro propojení systémů, napájení obvodů a jejich chlazení, jehož náročnost roste se zmenšováním prvků na obvodu.

2.4.1 Sběrnice

K propojování obvodů se používají sběrnice. Jejich unifikace umožňuje propojení různých druhů zařízení bez nutnosti řešit jejich kompatibilitu. Nejstarším typem je sběrnice paralelní. Její poslední rozšířenou variantou v PC byla sběrnice PATA a PCIx a v současnosti se zatím ještě používá na paměťové sběrnici a uvnitř čipů. Vzhledem k nevýhodám paralelních sběrnic se přešlo na systém sériových kanálů, např. PCI expres, HDMI, SATA.



Obr. 15. Šířka přenosových pásem v PC.

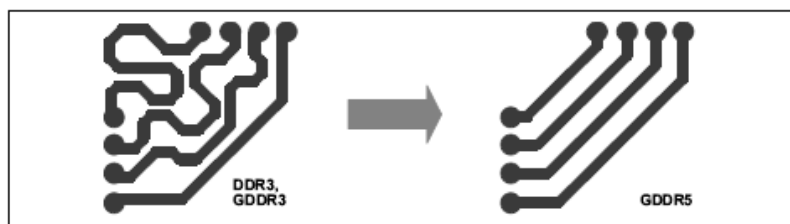
Na obrázku (Obr. 15) příklad sběrnice řešení současného PC s dedikovanou grafickou kartou. CPU je s hlavní pamětí propojen dvoj/troj kanálovou sběrnicí DDR3, grafický procesor je s pamětí zařízení propojen sběrnicí s časovou kompenzací rychlosti šíření signálu GDDR5 o šířce 64-1024 bitů dle výkonnosti GPU, CPU je s GPU propojen sběrnicí PCI express verze 2.0 nebo 3.0 s teoretickou rychlostí 8 nebo 16 GB/s na 16 spojích (lane).

Paralelní sběrnice

Umožňuje připojit více zařízení na jednu sběrnici, která přenáší v jednom okamžiku více bitů. Tím je zajištěna vyšší rychlost. V PC se od prvních 8-bitových ISA přešlo až na 32-bit PCIx. Nevýhodou těchto sběrnic je velké množství spojů (adresové, datové a řídicí spoje), rychlost sběrnice bývá omezena nejpomalejším připojeným zařízením, a aby byla zaručena časová dostupnost bitů, musí být všechny spoje stejně dlouhé. Také je problém s impedančním přizpůsobením sběrnice při připojení více zařízení různých délek.

Kompenzované paralelní sběrnice

Pomocí časovacích obvodů dokážou kompenzovat rozdílné délky spojů mezi obvody. Tato technologie se začala používat u pamětí pro grafické procesory (GPU). Umožňuje zjednodušit návrh plošných spojů a výrazně zmenšit zabranou plochu bez omezení rychlosti, jak je zobrazeno na layoutu spojů (Obr. 16) z manuálu paměti GDDR5 firmy Elpiada [4]. Nevýhodou zůstává omezení rychlosti nejpomalejším prvkem.



Obr. 16. Rozdíl layoutu pamětí DDR3/GDDR3 oproti pamětem GDDR5.

LVDS / 8b10b kódování

Začíná vytlačovat původní paralelní či jednoduché sériové sběrnice. Nejedná se už pouze o jednoduchou sběrnici, ale spíše o datový kanál, v němž jsou vedeny zároveň data, synchronizace, řídící informace a detekce chyb. V současné době se značně rozšiřuje právě kódování 8b/10b vedené po diferenciálním páru. Toto uspořádání umožňuje propojovat zařízení pomocí kanálů označovaných jako lane, které v sobě nesou všechny potřebné informace, které se dříve vedly zvláštními vodiči. Takto je řešena sběrnice PCI expres (konfigurace 1, 2, 4, 8, 16 nebo 32 lane), SATA, HDMI, DVI, Display Port, USB 3.0, Gigabit Ethernet. Tyto „sběrnice“ však již nejsou a nemohou být konstruovány jako point – multipoint, ale pouze jako point – point, tedy z jednoho bodu do jednoho bodu (zde se české slovo „sběrnice“ již jeví jako zavádějící).

2.4.2 Napájení

Při zmenšování struktur na čipu se také zmenšuje napájecí napětí obvodů. To je jednak nutné, protože menší struktura má nižší povolené napájecí napětí a také účelné, protože snížením napájení se zmenší svodové proudy a zmenší se energie potřebná k přepólování parazitních kapacit obvodů. Napájecí napětí se postupně snižovalo z 5V obvodů TTL a technologií NMOS, současné obvody mají napájení kolem 1,5V paměti, 1V procesorová jádra. Roste však napájecí proud a v současné době více než 50% pinů procesorů tvoří napájení. Přímě do procesorů se také přestěhovaly filtrační kondenzátory napájení.

2.4.3 Chlazení

Začíná být hlavním omezujícím prvkem výkonu obvodů. Zatím co si procesory řady x86 286, 386 vystačili s pasivním chlazením, od 486 se nároky na chlazení zvyšují a v současné době je výpočetní výkon obvodu takový, jaké množství tepla jsme schopni odvést tak, aby teplota čipu nepřesáhla kritickou hodnotu. Výkon tak již většinou není daný maximální použitelnou taktovací frekvencí ale schopností chlazení. Příkladem může být až

dvojnásobné zvýšení taktu CPU ale za použití extrémního chlazení. CPU na dvojnásobné frekvenci spolehlivě pracoval. U kancelářských PC se chlazení řeší jako vzduchové, u nejvýkonnějších serverů se podobně jako u mainframe serverů 70. tých let řešit jako vodní chlazení. Důvodem není jenom praktičnost, ale i rozměry. Pokud provozujeme paralelní úlohu, může být kritická i synchronizace vláken. Budeme-li například řešit neuronovou síť na více procesorech, můžou být výstupy jedné vrstvy vstupy na jiných procesorech. Budou-li dlouhé vzdálenosti spojů, bude i velké zpoždění při přenosu dat. Například počítače CRAY byly řešeny v kruhu, aby vzdálenost propojů byla co nejmenší.

Výkon obvodů se zhruba do uvedení procesorů PENTIUM 4 zvyšoval v souladu s Moorovým zákonem také zvyšováním frekvence taktu, ale poté se ztrátový výkon se zvyšující se frekvencí začal prudce zvyšovat a místo prostého zvyšování frekvence se začaly obvody paralelizovat a využívat složitější architektury. Pokud se nezmění technologie výroby klíčového prvku logických obvodů – v současnosti MOS transistoru, pravděpodobně se frekvence již výrazně ani zvyšovat nebude a klíčovým prvkem zvýšení výpočetního výkonu bude optimalizace procesorových jader a nárůst jejich počtu. Do optimalizace spadá i dynamické vypínání nepoužívaných obvodů, které jinak bez užitku produkují teplo.

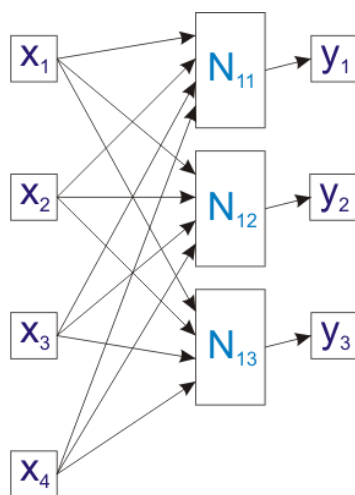
II. PRAKTICKÁ ČÁST

3 TECHNIKY SIMULACE NEURONOVÉ SÍTĚ

Softwarová simulace je řadou matematických a logických operací, které mají ve výsledku poskytnout podobný výsledek jako fyzická síť. Prakticky se jedná o operace násobení, sčítání, nelineárních funkcí, které jsou prováděny na polích vstupů, vah, výstupů a pomocných polí. Tyto operace vykonává jeden nebo více procesorů jako sérii instrukcí, při kterých pracuje s bloky alokované paměti představující vstupy, váhy, výstupy a pomocné pole. Tak alespoň můžeme vyjádřit simulaci perceptronu, který je náhradou, nikoli však úplnou, živého neuronu. Chování živého neuronu je naopak velmi složité, a to jak časovým průběhem, tak postupnou změnou parametrů. Přesná simulace by byla velmi složitá, zda-li vůbec možná.

3.1 Jednovrstvá neuronová síť bez rekurence

Je nejjednodušší variantou (Obr. 17). Cyklicky jsou načítány z matic váhy a vstupy, načtené hodnoty jsou vynásobeny a sumovány v pomocné proměnné. Ta je poté upravena nelineární funkcí a uložena do matice výstupu. Tím je nasimulována funkce jednoho perceptronu. Při více perceptronech je stejná činnost vykonána i pro ostatní. Největší zátěží tak činí datový tok matice vah, výpočetní potom funkce násobení a sumace.



Obr. 17. Jednovrstvá neuronová síť.

Datový tok pro každý cyklus je:

$D = (D_w + D_i + D_o) * N$, počet vah je dán $I * N$ což pro I vstupů, N perceptronů činí:

$D = I + N * I + N$ na cyklus. Hlavním datovým tokem je tok matice vah, který roste kvadraticky vzhledem k počtu vstupů a perceptronů ve vrstvě.

3.2 Jednovrstvá neuronová síť s rekurencí

Představitelem je například Hopfieldova síť. V tomto případě se počet vah zvyšuje ještě o počet rekurencí, kdy jsou na vstup těchto vah přivedeny hodnoty minulého cyklu (v případě diskrétního řešení).

3.3 Vícevrstvá neuronová síť

Datový a výpočetní výkon je daný součtem jednotlivých vrstev. Rozdíl může být v rekurenci, kdy jsou výstupy jedné vrstvy použity jako vstupy. Výstupy vrstev se stávají vstupem vrstev vyšších.

3.4 Optimalizace

Optimalizace může být provedena jako úprava algoritmu s ohledem na omezení výpočetních prostředků nebo rozdělením výpočtu na více procesorů. Algoritmickou úpravou může být například skalarizace výpočtu. Jde o techniku podobnou, jaká je použita u procesorů jako Instruction pipelining [14]. Ta může být využita k zlepšení vytížení výpočetních jednotek (MUL, ADD), kdy zatímco jednotka MUL provádí operaci v T_0 , jednotka ADD provádí operaci v T_{-1} , a ke zmenšení datového toku vah. Toho může být dosaženo výpočtem více po sobě jdoucích vstupů časově nebo různých vzorků v jednom cyklu (superskalárně).

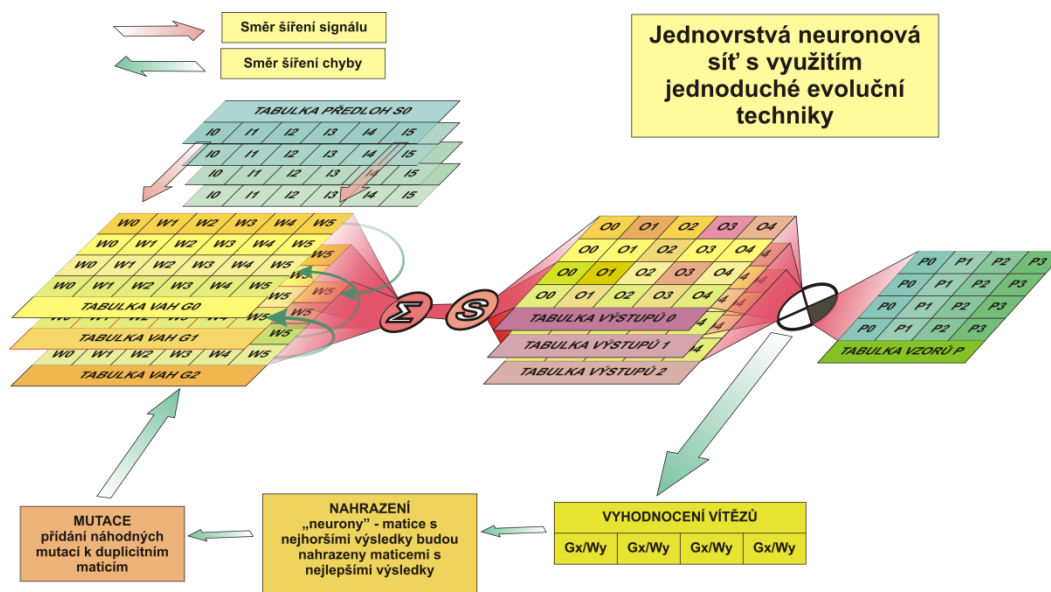
3.5 Náhrada rekurence

Superskalární provedení však zkomplikuje případnou rekurenci. Ta by byla zpožděná o n cyklů. Je tedy otázkou, k čemu rekurence slouží. Rekurence dělá z kombinační úlohy sekvenční, protože výstup pak nezáleží pouze na okamžité hodnotě vstupů ale i na hodnotách předchozích. Toho lze dosáhnout právě rekurencí, což je u neuronových sítí jediná možnost jak registrovat změny nebo si pamatovat stavy. U simulace můžeme podobného efektu dosáhnout předzpracováním dat na vstupu nebo mezi vrstvami pomocí metod jako je FFT, wavelet transformation, integrálním či diferenčním počtem nebo prostou pamětí hodnot.

4 SIMULACE NEURONOVÉ SÍTĚ POMOCI CPU A GPU

V této části jsou na simulovaných neuronových sítích identifikovány části nejvíce náročné na výpočetní výkon.

Programy jsou napsány v jazyce C/C++, je použito IDE Code:Block a Microsoft Visual C++ 2008 Express, CUDA SDK a CUDA toolkit. Bude uveden příklad jednovrstvé sítě s jednoduchým evolučním algoritmem (Obr. 18).



Obr. 18. Jednovrstvá neuronová síť s jednoduchou evoluční technikou.

4.1 CPU

Optimalizace kódu pro CPU je obtížná, protože ukládání částí dat a kódu do cache paměti si řídí sám processor. Výrazného zlepšení lze dosáhnout klíčovým slovem `register` u deklarace proměnných, kterou prioritizujete ukládání proměnných do registrů processoru, ev. řízením ukládání dat do nich napsáním klíčových částí v assembleru. Jejich doba přístupu je mnohem kratší než doba přístupu v paměti a běh programu se tím značně urychlí.

Část programu pro výpočet jednoho cyklu.

```
1 int vypocet_vrstvy_NN(int xBlock){
2     registered int tmp, nW, nN, nS, nE;
3     for (int nE=0;nE<ENTITIES;nE++){
4         for (int nS=0;nS<SAMPLE;nS++){
```

```

5          for (int nN=0;nN<LAYER_SIZE;nN++) {
6              tmp=0;
7              for (int nW=0;nW<LAYER_INPUTS;nW++) {
8                  tmp+=inBlock[nS][nW]*weight[nE][nN][nW];
9              }
10             outBlock[nE][nS][nN]=sigmoida(tmp);
11         }
12     }
13 }
14 return 0;
15 }

```

Počet výpočtů pro jednu vrstvu neuronů v tomto příkladu bude tedy dán násobkem počtu vstupů a počtu simulovaných neuronů. V každém výpočtu bude provedena funkce násobení a sčítání. Pro každý neuron bude provedena sigmoidní funkce. Dále tento blok výpočtů bude proveden pro všechny vstupní vzorky a poté ještě pro všechny simulované entity (jedince populace).

4.2 GPU-CUDA

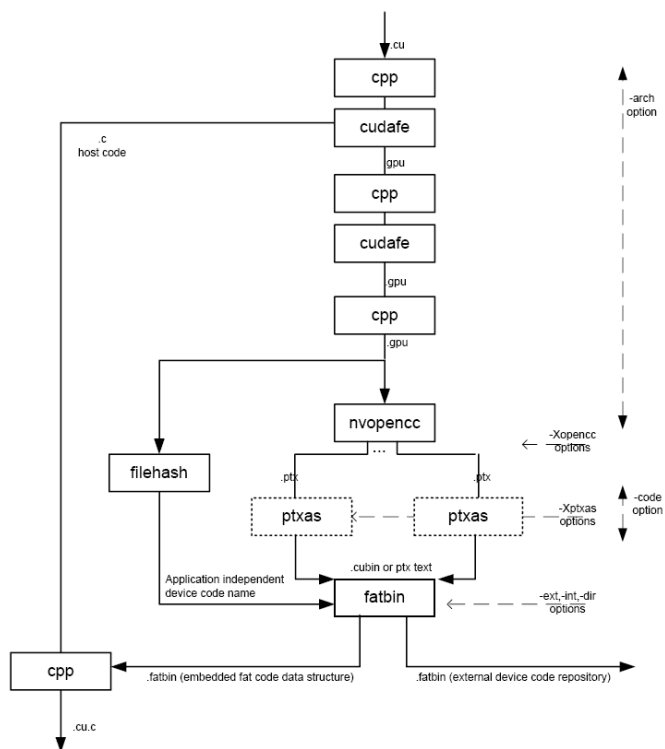
Hlavním rozdílem proti provedení na CPU je rozdělení úlohy do bloků, z nichž každý pak bude zpracováván jedním vláknem. Úkolem tedy je rozdělit úlohu na části tak, aby ji mohly vlákna vykonávat paralelně. Na konci jsou provedené úlohy synchronizovány.

Ve zdrojovém kódu jsou části pro CPU a GPU označeny klíčovými slovy uvedenými v tabulce (Tab. 3):

Tab. 3. Klíčová slova pro kompilátor nvcc.

Klíčové slovo	Běží na:	Spouštěné z:	
<code>__global__</code>	GPU	Host	
<code>__device__</code>	GPU	GPU	Při použití obou budou generovány verze pro CPU a GPU
<code>__host__</code>	Host	Host	

Kompilace zdrojového kódu pro CPU a GPU probíhá odlišně. Zdrojový kód může být napsán v běžném IDE, například MVC++. Zdrojový kód v jazyce C/C++ určený pro GPU, obvykle s příponou `.cu` je kompilován kompilátorem nvcc od NVIDIA.

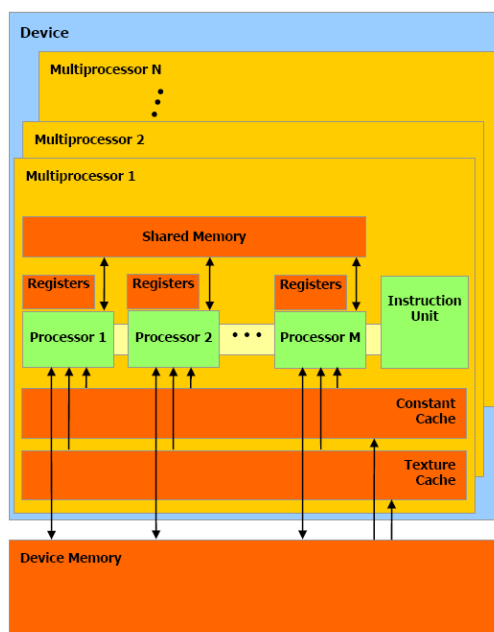


Obr. 19. nvcc compilation trajectory - dokumentace kompilátoru

nvcc, NVIDIA.

Tam je rozdělen na kód pro CPU, označovaném jako host, a kód pro GPU, jak je znázorněno na diagramu (Obr. 19). Kompilátor pro kód určený pro CPU využívá kompilátor `cl.exe` z MVC++ (podporovány verze 08 a 09). Spuštění programu na GPU probíhá tak, že CPU zkopíruje do paměti GPU data a program a poté dá instrukci k jeho spuštění na GPU.

Důležitou vlastností provozu aplikací na GPU je rozsáhlá možnost řízení toku dat, jejich ukládání a spouštění procesů. Jak jsem již výše uvedl, přesuny obsahu hlavní paměti do cache CPU L1 – L3 řídí procesor a je programátorem těžko ovlivnitelná. Naproti tomu ukládání dat do paměti GPU a paměti zařízení řídí plně řízena programátorem. Ten tak musí podle typu dat řídit ukládání a čtení do paměti zařízení, sdílené paměti, paměti textur nebo konstant, ev. registrů (Obr. 20).



Obr. 20. Architektura GPU, NVIDIA CUDA
programming guide.

Přístup do hlavní paměti (Global Memory, na obrázku jako Device Memory) se řídí podobně jako na CPU pomocí příkazů `cudaMalloc`, `cudaMemcpy` a `cudaFree`. Tato paměť není „kešovaná“ a její latence je 400-600 cyklů [11], což je hodně a lze ji snížit využitím sdruženého přístupu do této paměti (coalesced memory access) za dodržení určitých podmínek, což je obdoba prefetch X_n u hlavní paměti CPU.

Sdílená paměť, která je umístěná přímo na čipu GPU se alokuje přímo execution configuration [19]:

```
my_kernel<<<pocet_bloku, velikost_bloku, vel._sdilene_pameti_v_bytech>>>
```

a v kernelu definujeme:

```
__global__ void my_kernel () { //funkce
    __shared__ int s_data[velikost_v_bytech]; //deklarace pole
}
```

Samotné operace jsou poté provedeno v kernelu:

```
int bx = blockIdx.x; int by = blockIdx.y; //identifikace bloku
int tx = threadIdx.x; int ty = threadIdx.y; //identifikace vlákna
for (int k = 0; k < BLOCK_SIZE; ++k) {Csub += As[ty][k] * Bs[k][tx];}
//samotná operace v matici
```



```
__syncthreads(); //sysynchronizování vláken
```

Uvedená část odpovídá přibližně řádkům 3 – 12 předchozí ukázky vyjma sigmoidní funkci. Pro vlastní testování je vhodné použít již připravené funkce s balíku NVIDIA GPU COMPUTING SDK, kde jsou mimo těchto základních operací ještě ošetřeny chybové stavy a kolize. Při algoritmizaci úlohy na GPU je oproti CPU nutné znát architekturu procesoru, kvůli správnému určení velikosti bloků a počtu vláken. Neoptimalizovaná úloha poskytne mnohem horší výsledky.

Operace na multiprocesoru jsou typu SIMT. To znamená, že jedna instrukce je provedena na více vláknech. To mimo jiné znamená, že pokud se program má větvit, musí se rozdělit i blok vláken. Z tohoto důvodu je GPU vhodný prostředek pro paralelní výpočty nad maticemi dat, nikoli pro zpracování větvících se programů.

Algoritmus byl testován na konfiguraci:

CPU AMD Phenom II X2 3,11GHz

GPU NVIDIA GeForce 210

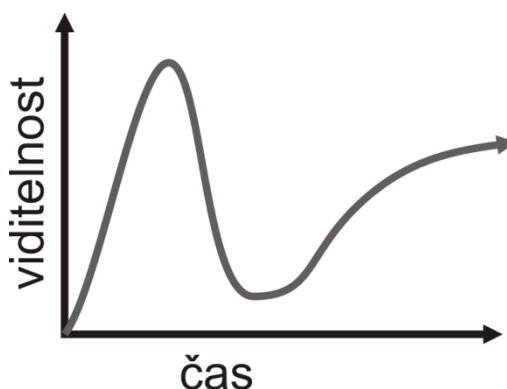
Programy provedené na GPU byly dle optimalizace přibližně 2x až 15x rychlejší než při provedení na CPU. Výsledky se lišili podle odladění obou způsobů a použitému vzorku z SDK. I když použitá grafická karta je nejnižší řadou podporující CUDA, svým paralelním výkonem předčí CPU (úlohu běžící na jednom jádře). Ostatní zdroje [21], [22] uvádějí přibližně 4x – 100x rychlejší zpracování na GPU od low-end (NVIDIA 9500, 210) po high-end (NVIDIA Tesla) oproti zpracování (v jednom vláknu) na běžném CPU (2 – 3 GHz). Pokud je tedy aplikace kritická na rychlost zpracování dat, vyplatí se převod programu (byť ne zcela triviální) z CPU na GPU.

5 VYUŽITÍ NEURONOVÝCH SÍTÍ

Neuronové sítě svůj nástup teprve čekají. Jejich současná kapacita je dostatečná na jednoduché aplikace, jako je rozpoznávání znaků, detekce obličejů, ale na složitější aplikace kde se již předpokládá „intelligence“ ještě nestačí. Počítač Watson od IBM byl spíše expertním systémem, tedy přístupem shora než stochastickým konekcionistickým systémem.

5.1 AI zima (AI winter)

Vývoj technologií často připomíná křivku (Obr. 21), kdy po počátečním nadšení z nástupu nové technologie nastupuje deziluze, způsobená nenaplněným očekáváním dosažených výsledků [13][9]. Poté nastává období klidu, kdy není o technologii zájem ze strany investorů a po jejím dozrání se postupně znovurozvíhá její vývoj následovaný dalšími investicemi.



Obr. 21. AI. winter, vývoj technologií.

Tato křivka trochu připomíná čtyři fáze tvůrčího procesu: tvůrčí nadšení, realizační vystřízlivění, potrestání nevinných a odměnění nezúčastněných.

Podmínkou masového nasazení pro trh je tedy celková zralost technologie. Ta musí zahrnovat nejen samotný prostředek AI, ale i jeho zasazení do stávajících systémů, například do operačního systému nebo prostředků pro vývoj aplikací.

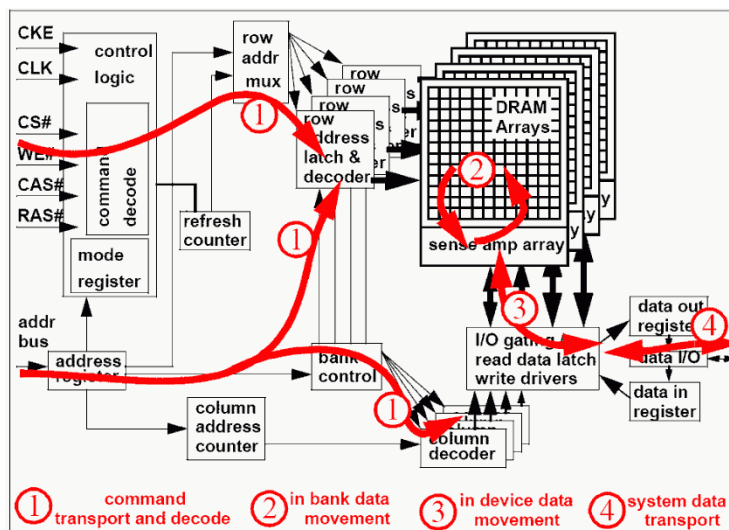
5.2 INTEGRACE AI DO SYSTÉMŮ

Aby bylo možné psát pro dané zařízení aplikace, je důležitá jeho integrace do operačního systému. To je patrné na příkladu využití GPU pro jiné účely. GPU je integrovaná do OS prostřednictvím ovladačů od výrobců, kteří nabízejí rozhraní Direct X nebo Open GL a jsou tak bohatě využívána. Naproti tomu využití CUDA od NVIDIA nebo FireStream od

ATI jsou již částečně integrovány do Direct X 11 jako DirectCompute, ale jejich využití není vysoké a je spíše na producentech software, zda-li uzpůsobí vlastní aplikace nabízeným prostředkům.

5.3 IMPLEMENTACE NEURONOVÝCH SÍTÍ DO HARDWARE

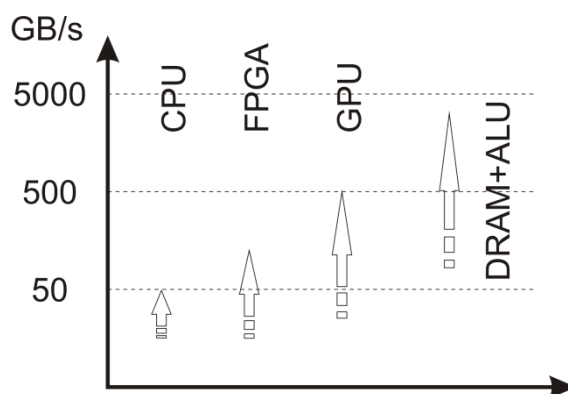
Z první části vyplývá, že nejméně optimální částí pro simulaci neuronové sítě je CPU, byť vícejádrový. Simulace na (GP)GPU je výrazným pokrokem, který může zvýšit výkon až o řád, ale stále ještě má strukturu pro řešení úloh mnohem složitějších, než jsou 2 aritmetické operace v simulaci neuronu a klíčová je rychlost načítání matice vah z paměti zařízení. Pro tento účel by tedy byl nejvhodnější speciální obvod, v němž by byla integrována jak paměť, tak aritmeticko-logická jednotka, jež by odstranila úzké hrdlo mezi pamětí a procesorem von Neumanovské architektury. Řešením by bylo doplnění stávajících pamětí (SDRAM, flash EEPROM) o logickou jednotku, která by prováděla jednoduché aritmetické operace na sběrnici paměti v šířce, která je u procesoru těžko fyzicky realizovatelná ale kterou disponují již stávající paměti. Podíváme-li se na vnitřní architekturu například paměti DDR3 o kapacitě 1Gbit (Obr. 22) [10], zjistíme, že čtení dat probíhá masivně paralelně.



Obr. 22. Datový tok v paměti SDRAM.

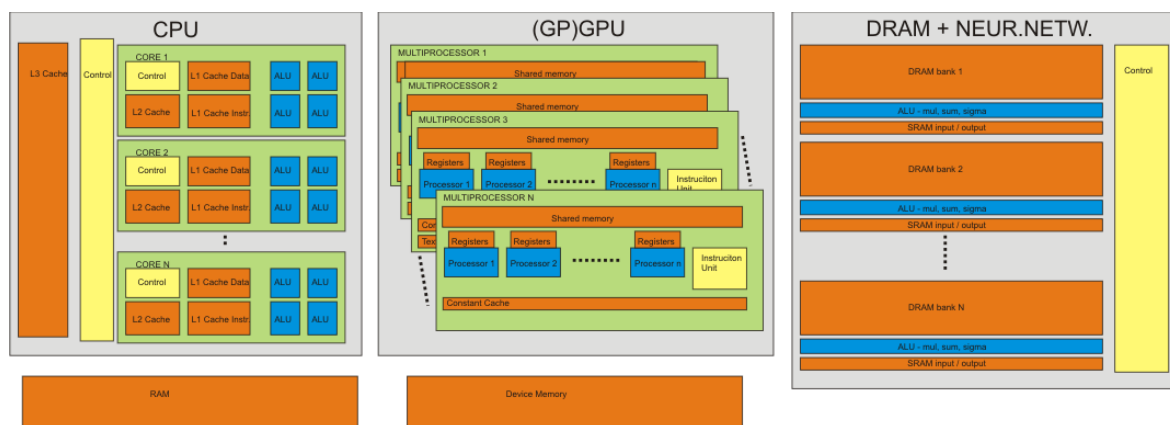
Matice paměti má architekturu 16k řádků, 8k sloupců v 8 bankách. Při čtecím nebo refresh cyklu se tak načítá všech 8k sloupců, při refresh navíc ve všech 8 bankách s časem cca. 50

ns / čtecí, 100 ns / refresh, takže během této doby se načte a zpětně uloží 8kb čtení / 8kB refresh dat. To znamená, že vnitřní tok po sobě jdoucích refresh cyklů je 80GB/s na jeden čip paměti střední třídy DDR3. Tato rychlost přesahuje rychlost paměťové sběrnice CPU a je na úrovni low-end grafické karty. Počet takovýchto čipů je však například pro paměť 4GB 32 kusů, což dává celkový datový tok asi 2,5 TB/s (pro refresh cyklus), tedy o řád výše než grafická karta střední třídy. Integrací jednoduché ALU na modul stávající běžné hlavní paměti bychom tak získali základ neuronové sítě (Obr. 24), jejíž výkon by byl jinak proveditelný jen na clusteru výkonných multiprocesorů high-end třídy (Obr. 23).



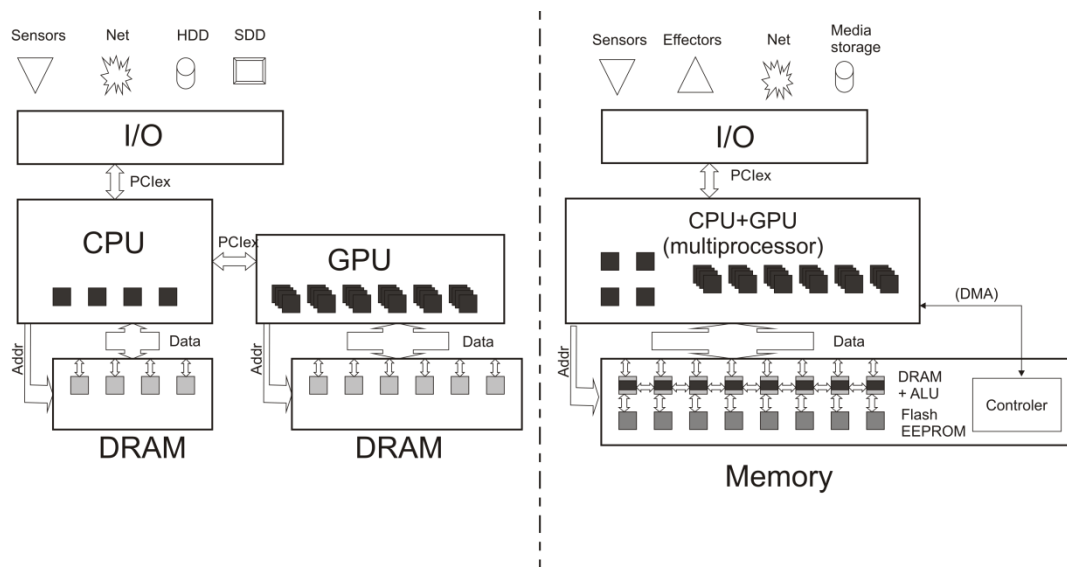
Obr. 23. Šířky pásma přístupu do DRAM.

V tomto případě je uvažován stávající paměťový čip bez optimalizace. Tou by bylo možné dosáhnout ještě dalšího zlepšení. Naopak by se muselo zlepšit chlazení čipů, které se v současnosti provádí jen u paměti high-end třídy.



Obr. 24. Možnosti lokalizace ALU a RAM.

Již probíhající kovergencí CPU a GPU a změnou přístupu k úložištím programů a dat by se mohla změnit architektura počítače (Obr. 25), jenž by již obsahovala DRAM s implementovanou vysoce paralelní ALU jako akcelerátorem simulace neuronových sítí.



Obr. 25. Změny v architektuře počítače.

5.4 ROZVOJ AI

V roce 2006 byl ve Švýcarsku institutem École Polytechnique v Lausanne spuštěn projekt, jehož cílem je pomocí reverzního inženýrství nasimulovat mozek savce. V projektu byla nasimulována velmi malá část mozku savce pomocí, v současné době jednoho z nejvýkonnějších superpočítačů, Blue Gene. Smyslem snažení je zálohování lidského mozku: Na to jsou dva přístupy, jeden je zkopírování existujících neuronů a jejich nahrazení umělou sítí. To je však těžko proveditelné vzhledem k velikosti sítě a vlastnostem přírodních neuronů. Navíc způsob, jakým by byl mozek naskanován nedává příliš smysl pro jeho simulaci. Průchodnější je propojení stávající sítě (mozku) s umělou a postupná kovergence. Mozek není jednou sítí ale má 2 hemisféry propojené cca. 200M spoji, což by mohlo být vodítkem pro složitost tohoto propojení. Mozek má navíc schopnost samoorganizace - neuroplasticity. Například v případě poranění nebo funkčních omezení částí mozku se centra přesouvají do nepoškozených a funkce se, alespoň částečně, obnovují. Podle některých studií by mohlo být úplné simulace dosaženo kolem roku 2050, což je však pouze hrubý odhad vycházející z Moorova zákona a kvalifikovaného odhadu náročnosti na simulaci.

Není zatím jasné, nakolik tyto technologie ovlivní běžný život. Jsou obavy, že člověk přestane být dominantní životní formou právě po spuštění systémů s umělou inteligencí. Intelligence je však nejasný pojem a je jen slovní hříčkou zda-li systémy, se svými algoritmy, jako google, facebook, ekonomické programy jsou nebo nejsou inteligentní.

Pokud usoudíme, že ano, pak můžeme mluvit v minulém čísle, protože značnou část našeho rozhodování již fakticky řídí. Značnou část podkladů k této práci byla vyhledána pomocí služeb google. Pokud označím tento systém za inteligentní, měl bych ho uvést jako spoluautora, protože výsledek ovlivnil. Mohl, pozitivně nebo negativně, pokud mu přisoudíme schopnost uvažování.

Další otázkou je také rychlost, jakou bude tento rozvoj probíhat. Živé organizmy se vyznačují schopností negentropie. To je opak entropie. Mají schopnost uspořádávat věci, částice, informace a snižovat tak jejich entropii – neurčitost. Pokud organizmus tuto vlastnost ztratí, je mrtvý. Například algoritmy vyhledávačů uspořádávají informace. Tuto vlastnost bychom mohli označit za známku života. Podle jedné z teorií bude vývoj pokračovat exponenciálně a v jenom okamžiku dosáhne singularity, tj. veškerá informační entropie zmizí. To je těžko uvěřitelné pro někoho, kdo pracoval ve státní správě. Datový tok v informačních sítích, které používáme, houstne, ale to neznamena, že se zvyšuje informovanost. Zvýšil se tok informací obecně, ale to neznamena, že se zvýšil to neduplicitních informací a je otázkou, zda je člověk také schopen větší informační toky přijmout. V takovém případě bychom mohli říct, že pouze roste entropie v našich informačních kanálech.

Celulární automaty můžou změnit pohled na evoluční techniky, genetické algoritmy a neuronové sítě. Při návrhu se pokoušíme o napodobení přírodních ekvivalentů. Co je snadno realizovatelné chemicko-biologickou cestou, nemusí být optimální pro digitální algoritmy. Může se tak v budoucnu ukázat, že evoluce, tak jak ji známe z biologického života, nemusí být optimální variantou pro vývoj deterministických systémů. Ta je založena na konečnosti existence jedince v populaci a existenci náhodné veličiny. Ryze deterministické systémy nám umožňují něco, co v systémech se stochastickou složkou nebylo možné. Například při losování nedostaneme stejná čísla, i kdybychom se snažili co nejvíce zachovat vstupní podmínky. Naproti tomu u ryze deterministických systémů, pokud zachováme vstupní podmínky a uměle nevložíme stochastickou složku – šum, bude výsledek vždy stejný. To umožňuje snáze zjišťovat závislost výstupu na vstupu nebo struktuře, nejsme omezeni pouze statistickými ukazateli.

ZÁVĚR

Hlavní cílem této práce je optimalizace hardware pro simulace neuronových sítí. Obecně se dá říci, že hlavním prvkem zvýšení efektivity může být odstranění úzkého hrdla mezi pamětí a procesorem von Neumannovské architektury jejich spojením do jednoho celku a redukce ALU na množinu potřebnou pro neuronové sítě. Použití GPGPU nebo FPGA ale také může být významným výkonnostním skokem. Pro předzpracování dat a ryze deterministické operace je vhodná dosavadní architektura s CPU nebo multiprocesorem, v současnosti představovaným GPGPU.

Memristor se může stát klíčovým prvkem při konstrukci neuronových sítí, jež by se více podobaly přírodním ekvivalentům. Jeho vývoj je v současné době na hranici realizovatelnosti, takže lze těžko odhadnout jeho klíčové parametry důležité pro vlastnosti případné neuronové sítě.

ZÁVĚR V ANGLIČTINĚ

The main purpose of this study is to optimize the hardware for the neural networks simulation. Generally, the main way of the increase in efficiency may be to eliminate the bottleneck between memory and von Neumann processor architecture out their merger into one unit and a reduction ALU instruction set. Using GPGPU or FPGA, but may also be a significant performance leap. For preprocessing and purely deterministic operation is suitable for the current architecture of the CPU or multiprocessors, currently represented by GPGPU.

Memristor can become a key element in the design of neural networks, which would be more like natural equivalents. Its development is currently on the border of possibility of producing, so it is difficult to estimate the key parameters important for the properties of any neural network.

SEZNAM POUŽITÉ LITERATURY

- [1] Kolektiv. *Průvodce po anatomii mozku a jeho funkcích*. Brno : Jota, 2009. 348 s. ISBN 978-80-7217-686-1.
- [2] HP. *Molecular crossbar latch*. Patent number: 6586965.
- [3] SUNG, Hyun Jo, et al. *Nanoscale Memristor Device as Synapse in Neuromorphic Systems*. Department of Electrical Engineering and Computer Science, University of Michigan. Dostupné z WWW: <http://www.eecs.umich.edu/~wluee/LuJo_MemristorSynapse_NL2010.pdf>
- [4] ELPIDA. *Introduction To GDDR5 SGRAM*. USER'S MANUAL. Dostupné z WWW: <www.elpida.com/pdfs/E1600E10.pdf>.
- [5] *Gordon Moore official page* [online]. 2011 [cit. 2011-05-30]. Dostupné z WWW: <<http://www.intel.com/technology/mooreslaw/>>.
- [6] MAŘÍK V., ŠTĚPÁNKOVÁ O., LAŽANSKÝ J. *Umělá inteligence IV*. Academia, Praha. ISBN 80-200-1044-0, 2004
- [7] *Neurochemie*. Univerzita Karlova, přednášky. Dostupné z WWW: <http://web.natur.cuni.cz/fyziol/odd_neuro/neurochem_2.pdf>
- [8] AZVEDO, et al. Equal numbers of neuronal and non-neuronal cells make the human brain an isometrically scaled-up primate brain. *Journal of Comparative Neurology* [online]. 2009, 513, 532-541, [cit. 2011-05-30]. Dostupný z WWW: <http://www.suzanaherculanohouzel.com/reprints/Azevedo2009JCompNeurol_HumanBrainComposition.pdf>.
- [9] AI winter. In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 28 December 2005 , last modified on 16 May 2011 [cit. 2011-05-30]. Dostupné z WWW: <http://en.wikipedia.org/wiki/AI_winter>.
- [10] WANG, David Tawei . *Modern DRAM memory systems: performance analysis and scheduling algorithm*. Faculty of the Graduate School of the University of Maryland, 2005. 248 s. Dizertační práce. University of Maryland. Dostupné z WWW: <www.ece.umd.edu/~blj/papers/thesis-PhD-wang--DRAM.pdf>.
- [11] ZAORÁLEK, Lukáš. *Root.cz* [online]. 2009 [cit. 2011-05-30]. CUDA: více o sdruženém přístupu do globální paměti. Dostupné z WWW: <<http://www.root.cz/clanky/cuda-vice-o-sdruzenem-pristupu-do-globalni-pameti>>.

- [12] DIAS, F. M., ANTUNES, A., MOTA, A. M. *Commercial hardware for artificial neural networks: A Survey*. IFAC. 2002. 8s. Dostupné z WWW: <http://cee.uma.pt/morgado/Down/HardwareNN_final2.PDF>.
- [13] HENDLER, James. *Avoiding Another AI Winter*. Published by the IEEE Computer Society 2008, 1541-1672/08,
- [14] ROJAS, RAÚL. *Neural Networks*. Berlin: Springer-Verlag, 1996. 509s, ISBN 3-540-60505-3.
- [15] XILINX, *Virtex-7 FPGA Family*,. [online]. 2011 [cit. 2011-05-30]. Dostupné z WWW: <<http://www.xilinx.com/products/silicon-devices/fpga/virtex-7/index.htm>>.
- [16] GERSTNER, Wulfram, KISTLER, Werner M. *Spiking Neuron Models*. Cambridge: University Press, 2002. ISBN 0 521 89079 9.
- [17] ZELINKA, Ivan. *Umělá inteligence - hrozba nebo naděje?*. Praha : BEN - technická literatura, 2003. 144 s. ISBN 80-7300-068-7.
- [18] Intel. High-K metal gate. *Intel pressroom* [online]. 2010, [cit. 2011-05-30]. Dostupný z WWW: <http://download.intel.com/pressroom/video/High-k_Metal_Gate_animation_640X480.wmv>.
- [19] ZAORÁLEK, Lukáš. *Root.cz* [online]. 2009 [cit. 2011-05-30]. CUDA: optimalizace přístupu do globální paměti. Dostupné z WWW: <<http://www.root.cz/clanky/cuda-optimalizace-pristupu-do-globalni-pameti>>.
- [20] ALTERA, *Stratix V FPGAs: Built for Bandwidth*,. [online]. 2011 [cit. 2011-05-30]. Dostupné z WWW: <<http://www.altera.com/products/devices/stratix-fpgas/stratix-v/stxv-index.jsp>>.
- [21] NeuroDimension inc., *NeuroSolutions CUDA Add-on*,. [online]. 2011 [cit. 2011-05-30]. Dostupné z WWW: <<http://www.neurosolutions.com/products/cuda/>>.
- [22] CodeProject., *A Neural Network on GPU*,. [online]. 2011 [cit. 2011-05-30]. Dostupné z WWW: <<http://www.codeproject.com/KB/graphics/GPUNN.aspx>>.

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

ALU	Arithmetic Logic Unit, obvody vykonávající aritmetické a logické operace.
(GP)GPU	(General-purpose computing on) Graphics Processing Units, (provádění obecných výpočtů na) grafickém procesoru.
CA	Cellular automat, buněčný automat.
CPU	Central Precessor Unit, mikroprocessor, obvod zahrnující aritmeticko-logickou jednotku a obvody řízení.
CUDA	Compute Unified Device Architecture, jednotná architektura pro obecné výpočty na zařízeních GPU NVIDIA.
DRAM	Dynamická RAM, její obsah je uložen v náboji na kondenzátoru, má destruktivní čtení.
DSP	Digital Signal Precessor, obvod podobný mikroprocesoru určený ke zpracovávání signálů, k čemuž je vybaven sadou speciálních instrukcí.
EPROM	Erasable Programable Random Only Memory, mazatelná paměť která jinak slouží pouze k čtení hodnot.
IDE	Integrated Developement Enviroment, integrované vývojové prostředí zahrnující obvykle textový editor kódu a kompilátor.
LVDS	Low Voltage Difential Signal, signál s nízkým napětím na symetrickém páru.
RAM	Random Access Mermory, paměť s náhodným přístupem.
SDK	Software Development Kit, softwarový vývojový balík, sada programů, utilit a příkladů kódu pro zrychlení a usnadnění vývoje aplikací
SRAM	Statická RAM, obsah je uložen ve stavu klopného obvodu.

SEZNAM OBRÁZKŮ

Obr. 1. Přenos impulsu neuronem.	13
Obr. 2. Počet neuronů v jednotlivých částech mozku, Azevedo et al. (2009).	14
Obr. 3. Přenosové funkce spojení více neuronů.	15
Obr. 4. Perceptron.	15
Obr. 5. Třívrstvá neuronová síť s jednou skrytou vrstvou.	17
Obr. 6.: Neuronová síť s rekurencí.	17
Obr. 7: Hopfieldova síť.	18
Obr. 8. Algoritmy inspirované přírodou.	18
Obr. 9. Rozdělení umělých neuronových sítí.	20
Obr. 10. Ukázky z prezentace využití High-K izolačních vrstev firmy INTEL.	25
Obr. 11. Memristorová síť a memristor.	26
Obr. 12. Molecular crossbar latch z patentu firmy HP.	27
Obr. 13. Vnitřní struktura paměti DRAM, Glogger at en.wikipedia.	28
Obr. 14. Schéma optické varianty neuronové sítě.	30
Obr. 15. Šířka přenosových pásem v PC.	31
Obr. 16. Rozdíl layoutu paměti DDR3/GDDR3 oproti pamětem GDDR5.	32
Obr. 17. Jednovrstvá neuronová síť.	35
Obr. 18. Jednovrstvá neuronová síť s jednoduchou evoluční technikou.	37
Obr. 19. nvcc compilation trajectory - dokumentace kompilátoru nvcc, NVIDIA.	39
Obr. 20. Architektura GPU, NVIDIA CUDA programming guide.	40
Obr. 21. AI. winter, vývoj technologií.	42
Obr. 22. Datový tok v paměti SDRAM.	43
Obr. 23. Šířky pásma přístupu do DRAM.	44
Obr. 24. Možnosti lokalizace ALU a RAM.	44
Obr. 25. Změny v architektuře počítače.	45

SEZNAM TABULEK

Tab. 1. Výběr obvodů FPGA firmy Xilinx.....	23
Tab. 2: Fyzické řešení neuronových sítí.....	24
Tab. 3. Klíčová slova pro kompilátor nvcc.....	38

SEZNAM PŘÍLOH

P I: Ukázky zdrojových kódů

PŘÍLOHA P I: UKÁZKY ZDROJOVÝCH KÓDŮ

Zpracování násobení matic na CPU

```
16 void cpu_mul(const int n, const float *A, const float *B, float *C) {
17     float sum = 0;
18     for (int i = 0; i < n; ++i) {
19         for (int j = 0; j < n; ++j) {
20             sum = 0;
21             for (int k = 0; k < n; ++k) {
22                 sum += A[i * n + k] * B[k * n + j];
23             }
24             C[i * n + j] = sum;
25         }
26     }
27 }
```

Zpracování násobení matic na GPU (zkráceno). Definice kernelu, bude zpracován na GPU (klíčové slovo `__global__`):

```
28 __global__ void matrixMul(int n, float* A, float* B, float* C) {
29     int bx = blockIdx.x; int by = blockIdx.y;
30     int tx = threadIdx.x; int ty = threadIdx.y;
31     int aBegin = n * BLOCK_SIZE * by; //začátek matice
32     int aEnd = aBegin + n - 1; //konec matice
33     float Csub = 0;
34     for (int a = aBegin, b = BLOCK_SIZE * bx; a <= aEnd; a +=
BLOCK_SIZE, b += BLOCK_SIZE * n) {
35         __shared__ float As[BLOCK_SIZE][BLOCK_SIZE];
        //deklarace matice ve sdílené paměti GPU
36         __shared__ float Bs[BLOCK_SIZE][BLOCK_SIZE];
37         As[ty][tx] = A[a + n * ty + tx];
38         Bs[ty][tx] = B[b + n * ty + tx];
39         __syncthreads();
40         for (int k = 0; k < BLOCK_SIZE; ++k) {
41             Csub += As[ty][k] * Bs[k][tx]; //výpočet
42         }
43         __syncthreads();
44     }
45     int c = n * BLOCK_SIZE * by + BLOCK_SIZE * bx;
46     C[c + n * ty + tx] = Csub;
47 }
```

Definice funkce, bude spuštěna na CPU, zkopíruje matice do paměti GPU, pustí kernel, poté zkopíruje data z paměti GPU zpět do CPU.

```
48 void cuda_mul(const int n, const float *hostA, const float *hostB, float
    *hostC) {
49     const int size = n * n * sizeof(float);
50     float *devA, *devB, *devC;
51     cudaMalloc((void**)&devA, size); //alokace paměti GPU, podobná jako
    u CPU
52     cudaMalloc((void**)&devB, size);
53     cudaMalloc((void**)&devC, size);
54     cudaMemcpy(devA, hostA, size, cudaMemcpyHostToDevice); //kopírování
    z paměti CPU do paměti GPU
55     cudaMemcpy(devB, hostB, size, cudaMemcpyHostToDevice);
56     dim3 threads(BLOCK_SIZE, BLOCK_SIZE);
57     dim3 grid(n / threads.x, n / threads.y);
58                                     //call kernel matrixMul
59     matrixMul<<<grid, threads>>>(n, devA, devB, devC); //vlastní
    spuštění programu na GPU
60     cudaMemcpy(hostC, devC, size, cudaMemcpyDeviceToHost); //
    zkopírování výsledku zpět do paměti CPU
61     cudaFree(devA); // uvolnění paměti
62     cudaFree(devB);
63     cudaFree(devC);
64 }
```