Výkonnost, RVP

INP - cvičení

Zdeněk Vašíček

2025

Relativní výkonnost

- Zvolíme referenční počítač
- Zvolíme sadu testovacích úloh (tzv. benchmarky)
- Spočítáme relativní doby výpočtu úloh na daném počítači (tj. vzhledem k referenčnímu počítači)

- Z relativních dob můžeme určit a použít ke srovnání:
  - aritmetický průměr
  - vážený průměr
  - geometrický průměr

- Obvykle je používán **geometrický průměr** (viz sady testovacích úloh SPEC, Whetstone, Dhrystone, …)

Sada testovacích úloh SPEC89

Benchmarks	Type	Application Description
001.gcc	INT	GNU C compiler
008.espresso	INT	PLA optimizing tool
013.spice2g6	FP	Circuit simulation and analysis
015.doduc	FP	Monte Carlo simulation
020.nasa7	FP	Seven floating-point kernels
022.li	INT	LISP interpreter
023.eqntott	INT	Conversions of equations to truth table
030.matrix300	FP	Matrix solutions
042fpppp	FP	Quantum chemistry application
047.tomcatv	FP	Mesh generation application

INT = Integer intensive benchmark.
FP = Double-precision floating-point benchmark.

$$\mathrm{SPEC}_{\mathrm{int89}} = \sqrt[4]{\prod_{i=1}^{4} t_{rel_i}}$$

$$\mathrm{SPEC}_{\mathrm{fp89}} = \sqrt[6]{\prod_{i=1}^{6} t_{rel_i}}$$

$$t_{rel_i} = \frac{t_{ref_i}}{t_{act_i}}$$

Srovnání výkonnosti dvou systémů s využitím SPEC89

$SPEC_{int89}$ (4 programy), $SPEC_{fp89}$ (6 programů)

Počítač / Úloha	VAX11/780 (s)	DEC3100 (s)	SPARC (s)	DEC3100 (relativní)	SPARC (relativní)
GCC	1482	145	138,9	10,22	10,67
Espresso	2266	194	254	11,68	8,92
Li	6206	480	689,5	12,93	9,08
Eqntott	1101	99	113,5	11,12	9,7
Spice	23951	2500	2875,5	9,58	8,33
Doduc	1863	208	374,1	8,96	4,98
NASA7	20093	1646	2308,2	12,21	8,71
Matrix300	4525	749	409,3	6,04	11,06
Fpppp	3038	292	387,2	10,4	7,85
Tomcatv	2649	260	469,8	10,19	5,64
	Referenční počítač		**SPEC-FX**	**11,45**	**9,57**
			SPEC-FP	**9,36**	**7,49**

$SPEC_{INT}$

$SPEC_{FP}$

DEC3100 je výkonnější než SPARC

Sada testovacích úloh SPEC CPU2006

http://www.spec.org/cpu2006/results/

SPECINT 2006 (12 benchmarků)

Benchmark	Language	Application Area
400.perlbench	C	Programming Language
401.bzip2	C	Compression
403.gcc	C	C Compiler
429.mcf	C	Combinatorial Optimization
445.gobmk	C	Artificial Intelligence: Go
456.hmmer	C	Search Gene Sequence
458.sjeng	C	Artificial Intelligence: chess
462.libquantum	C	Physics / Quantum Computing
464.h264ref	C	Video Compression
471.omnetpp	C++	Discrete Event Simulation
473.astar	C++	Path-finding Algorithms
483.xalancbmk	C++	XML Processing

SPECFP 2006 (17 benchmarků)

Benchmark	Language	Application Area
410.bwaves	Fortran	Fluid Dynamics
416.gamess	Fortran	Quantum Chemistry.
433.milc	C	Physics / Quantum Chromodynamics
434.zeusmp	Fortran	Physics / CFD
435.gromacs	C, Fortran	Biochemistry / Molecular Dynamics
436.cactusADM	C, Fortran	Physics / General Relativity
437.leslie3d	Fortran	Fluid Dynamics
444.namd	C++	Biology / Molecular Dynamics
447.dealII	C++	Finite Element Analysis
450.soplex	C++	Linear Programming, Optimization
453.povray	C++	Image Ray-tracing
454.calculix	C, Fortran	Structural Mechanics
459.GemsFDTD	Fortran	Computational Electromagnetics
465.Tonto	Fortran	Quantum Chemistry
470.lbm	C	Fluid Dynamics
481.wrf	C, Fortran	Weather
482.sphinx3	C	Speech recognition

$$\mathrm{SPEC}_{\mathrm{int2006}} = \sqrt[12]{\prod_{i=1}^{12} t_{rel\,i}} \qquad t_{rel_i} = \frac{t_{ref_i}}{t_{act_i}}$$

$$\mathrm{SPEC}_{\mathrm{fp\,2006}} = \sqrt[17]{\prod_{i=1}^{17} t_{rel\,i}} \qquad t_{rel_i} = \frac{t_{ref_i}}{t_{act_i}}$$

Sada testovacích úloh SPEC CPU2017

http://www.spec.org/cpu2017/results/

SPEC CPU2017 - 43 benchmarků rozdělených do 4 skupin

- SPECspeed – zaměřeno na dobu zpracování
- SPECrate – zaměřeno na propustnost

SPECrate 2017 Integer	SPECspeed 2017 Integer	Language	Application Area
500.perlbench_r	600.perlbench_s	C	Perl interpreter
502.gcc_r	602.gcc_s	C	GNU C compiler
505.mcf_r	605.mcf_s	C	Route planning
520.omnetpp_r	620.omnetpp_s	C++	Discrete Event simulation - computer network
523.xalancbmk_r	623.xalancbmk_s	C++	XML to HTML conversion via XSLT
525.x264_r	625.x264_s	C	Video compression
531.deepsjeng_r	631.deepsjeng_s	C++	Artificial Intelligence: alpha-beta tree search (Chess)
541.leela_r	641.leela_s	C++	Artificial Intelligence: Monte Carlo tree search (Go)
548.exchange2_r	648.exchange2_s	Fortran	Artificial Intelligence: recursive solution generator (Sudoku)
557.xz_r	657.xz_s	C	General data compression

SPECrate 2017 Floating Point	SPECspeed 2017 Floating Point	Language	Application Area
503.bwaves_r	603.bwaves_s	Fortran	Explosion modeling
507.cactuBSSN_r	607.cactuBSSN_s	C++, C, Fortran	Physics: relativity
508.namd_r		C++	Molecular dynamics
510.parest_r		C++	Biomedical imaging: optical tomography with finite elements
511.povray_r		C++, C	Ray tracing
519.lbm_r	619.lbm_s	C	Fluid dynamics
521.wrf_r	621.wrf_s	Fortran, C	Weather forecasting
526.blender_r		C++, C	3D rendering and animation
527.cam4_r	627.cam4_s	Fortran, C	Atmosphere modeling
	628.pop2_s	Fortran, C	Wide-scale ocean modeling (climate level)
538.imagick_r	638.imagick_s	C	Image manipulation
544.nab_r	644.nab_s	C	Molecular dynamics
549.fotonik3d_r	649.fotonik3d_s	Fortran	Computational Electromagnetics
554.roms_r	654.roms_s	Fortran	Regional ocean modeling

Příklady benchmarků

SPEC CPU®2017 Floating Point Rate Result

Dell Inc.
Dell PowerEdge R540 (Intel Xeon Bronze 3204, 1.90 GHz)

SPECrate®2017_fp_base = 49.4

SPECrate®2017_fp_peak = 50.1

CPU2017 License:	55	**Test Date:**	Sep-2019
Test Sponsor:	Dell Inc.	**Hardware Availability:**	Apr-2019
Tested by:	Dell Inc.	**Software Availability:**	May-2019

Benchmark result graphs are available in the PDF report.

Hardware

CPU Name:	Intel Xeon Bronze 3204
Max MHz:	1900
Nominal:	1900
Enabled:	12 cores, 2 chips
Orderable:	1,2 chips
Cache L1:	32 KB I + 32 KB D on chip per core
L2:	1 MB I+D on chip per core
L3:	8.25 MB I+D on chip per chip
Other:	None
Memory:	384 GB (12 x 32 GB 2Rx4 PC4-2933P-R, running at 2133)
Storage:	480 GB SATA SSD
Other:	None

Software

OS:	Ubuntu 18.04.2 LTS 4.15.0-45-generic
Compiler:	C/C++: Version 19.0.4.227 of Intel C/C++ Compiler Build 20190416 for Linux; Fortran: Version 19.0.4.227 of Intel Fortran Compiler Build 20190416 for Linux
Parallel:	No
Firmware:	Version 2.2.11 released Jun-2019
File System:	ext4
System State:	Run level 5 (multi-user)
Base Pointers:	64-bit
Peak Pointers:	64-bit
Other:	None
Power Management:	--

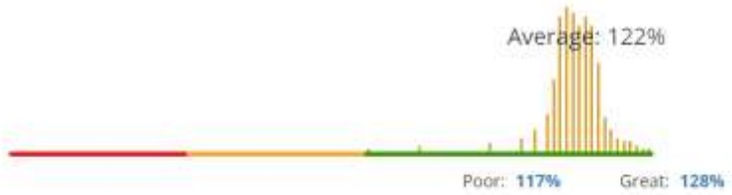
Results Table

Benchmark	Base							Peak						
	Copies	Seconds	Ratio	Seconds	Ratio	Seconds	Ratio	Copies	Seconds	Ratio	Seconds	Ratio	Seconds	Ratio
503.bwaves_r	12	**757**	**159**	752	160			12	754	160	**756**	**159**		
507.cactuBSSN_r	12	428	35.5	**429**	**35.4**			12	**443**	**34.3**	428	35.5		
508.namd_r	12	**355**	**32.1**	355	32.2			12	351	32.4	**359**	**31.7**		
510.parest_r	12	928	33.8	**929**	**33.8**			12	931	33.7	**931**	**33.7**		
511.povray_r	12	**584**	**48.0**	584	48.0			12	**502**	**55.8**	500	56.0		
519.lbm_r	12	315	40.2	**315**	**40.2**			12	**315**	**40.2**	315	40.2		
521.wrf_r	12	**480**	**56.1**	478	56.2			12	469	57.4	**469**	**57.3**		
526.blender_r	12	**489**	**37.4**	487	37.5			12	488	37.5	**490**	**37.3**		
527.cam4_r	12	506	41.5	**506**	**41.5**			12	486	43.1	**487**	**43.1**		
538.imagick_r	12	**304**	**98.3**	302	98.8			12	**306**	**97.6**	301	99.0		

Další benchmarky

Další benchmarky

- SPEC se zaměřením na MPI, OpenACC a OpenMP
- 3DMark
- Userbenchmark
  - Pozor na validitu výsledků, může být i bias - např u userbenchmark se řešila otázka, zda nezvýhodňuje Intel oproti AMD



Měření výkonnosti

$$t_e = N_I \cdot CPI \cdot T_{clk} = \frac{N_I \cdot CPI}{f_{clk}}$$

- t_e – doba výpočtu (vykonávání programu)
- N_I – počet instrukcí
- T_{clk} – perioda hodinového signálu ($T_{clk} = 1/ f_{clk}$)
- CPI – počet hodinových cyklů na instrukci (**C**ycles **P**er **I**nstruction)

$$P_{MIPS} = \frac{N_I}{t_e \cdot 10^6} = \frac{f_{clk}}{CPI \cdot 10^6}$$ (instrukcí za sekundu v milionech)

$$P_{MFLOPS} = \frac{N_{FPI}}{t_e \cdot 10^6}$$ (FP instrukcí za sekundu v milionech)

$$\text{relativní } P_{\text{MIPS}} = t_{\text{e ref. poč.}} \cdot P_{\text{MIPS ref. poč.}} / t_{\text{e měř. poč.}}$$

Příklad 1

Na počítačích A a B jsme spustili program.

Naměřili jsme t_A=10s, t_B=15s.

O kolik % je A výkonnější než B?

$$\frac{P_A}{P_B} = \frac{t_B}{t_A} = \frac{15}{10} = 1{,}5$$

Příklad CPI vs doba výpočtu

Optimalizující kompilátor odstraní 50% instrukcí ALU, ostatní instrukce ponechá. T_{clk}=2ns.

Instr.	Četnost	CPI	Opt.
ALU	43 %	1	21.5%
Load	21 %	2	21%
Store	12 %	2	12%
Jmp	24 %	2	24%
	100 %		78.5 %

1. Vypočtěte $CPI_{původ}$, CPI_{opt}, $P_{MIPSpův}$, $P_{MIPSopt}$.

$$CPI_{původ} = 0.43 \cdot 1 + 0.21 \cdot 2 + 0.12 \cdot 2 + 0.24 \cdot 2 \cong 1.57$$

$$CPI_{opt} = \frac{0.215 \cdot 1 + 0.21 \cdot 2 + 0.12 \cdot 2 + 0.24 \cdot 2}{0.758} \cong 1.73$$

$$P_{MIPS,původ} = \frac{f_{clk}}{CPI_{původ} \cdot 10^6} = \frac{1}{CPI_{původ} \cdot T_{clk} \cdot 10^6} \cong 318.4$$

$$P_{MIPS,opt} = \frac{f_{clk}}{CPI_{opt} \cdot 10^6} = \frac{1}{CPI_{opt} \cdot T_{clk} \cdot 10^6} \cong 289.0$$

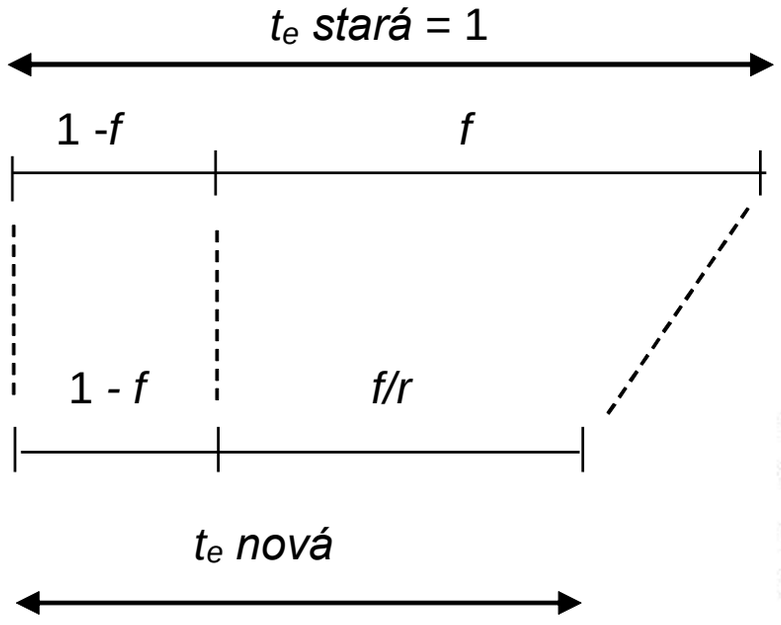
$$t_{e,původ} = N_I \cdot CPI_{původ} \cdot T_{clk} \cong 3.14 \cdot 10^{-9} \cdot N_I$$

$$t_{e,opt} = N_I \cdot CPI_{opt} \cdot T_{clk} \cong 2.7 \cdot 10^{-9} \cdot N_I$$

2. Souhlasí poměr P_{MIPS} a t_e? Co je tu zvláštní? Který parametr je pro nás důležitější?

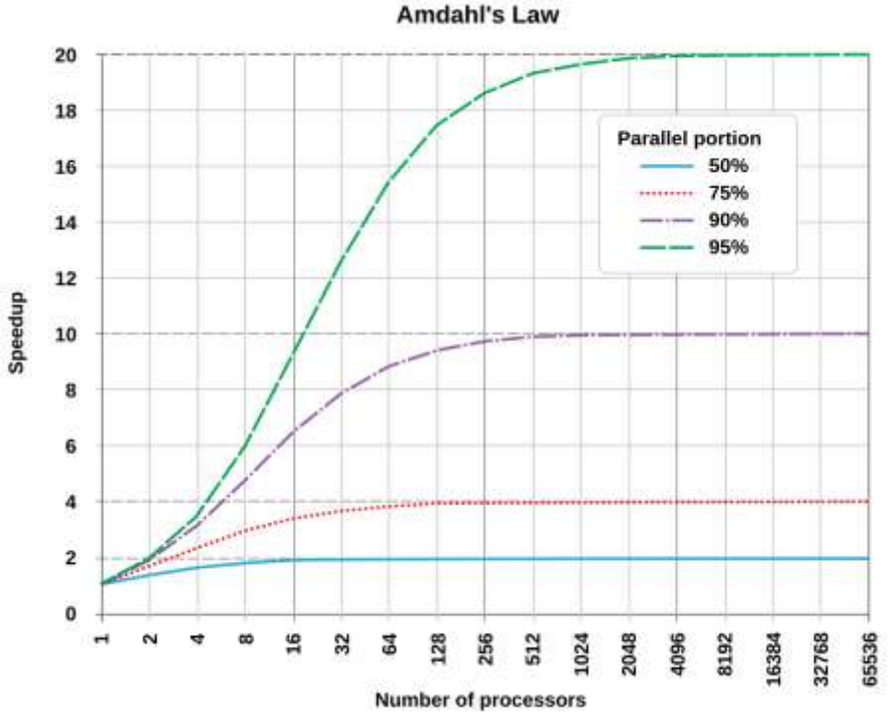
Amdahlův zákon

Část f můžeme urychlit r-krát.



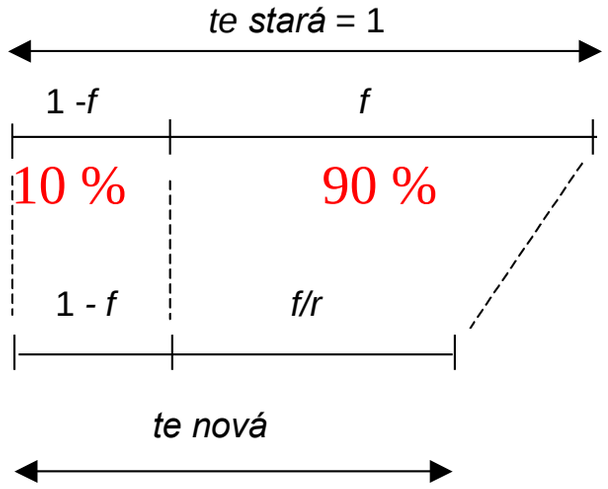
Celkové zrychlení je

$$V_r = \frac{t_{e,stará}}{t_{e,nová}} = \frac{1}{(1-f) + \frac{f}{r}}$$



Příklad 3

Cache je 5x rychlejší než hlavní paměť a využívá se z 90% výpočetního času. Jaké se dosáhne zrychlení zavedením této cache?



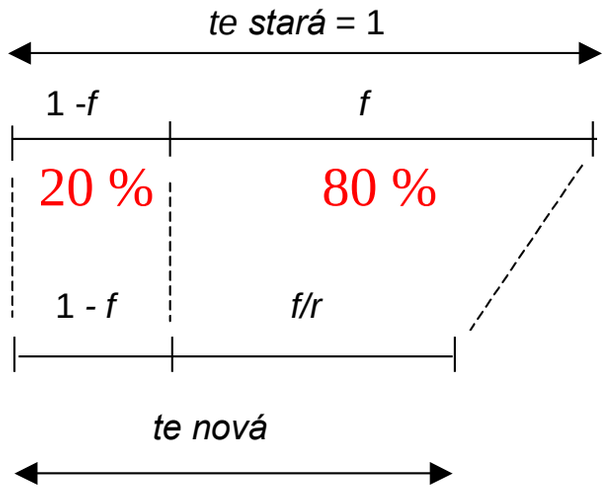
$f = 0.9$

$r = 5$

$$V_r = \frac{t_1}{t_2} = \frac{1}{(1-f)+\frac{f}{r}} = \frac{1}{0.1+\frac{0.9}{5}} = 3.57$$

Příklad 4

Máme sekvenční program. Pro 80% kódu jsme schopni jej přepsat na paralelní s tím, že bude ideálně využívat 4 jádra CPU. Jaké je zrychlení?



$$f = 0.8$$

$$r = 4$$

$$V_r = \frac{1}{(1-f)+\frac{f}{r}} = \frac{1}{0.2+\frac{0.8}{4}} = 2.5$$

Výhodnost koupě nového CPU

Procesor je využit z 50%, zbytek času čeká na I/O operace. Cena procesoru tvoří 1/3 ceny počítače. Je výhodné koupit 5x rychlejší CPU za 5x vyšší cenu?

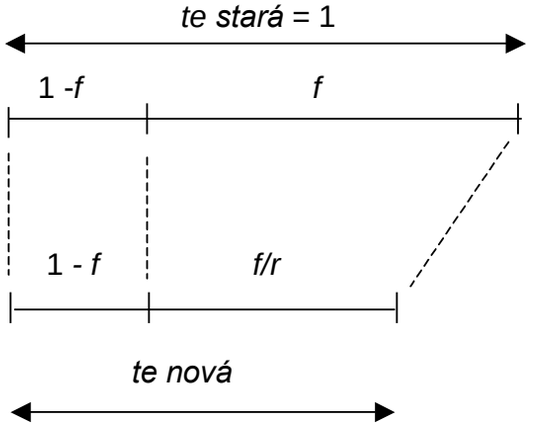
$f = 0.5, \quad r = 5$

zrychlení:

$$V_r = \frac{1}{(1-f) + \frac{f}{r}} = \frac{1}{(1-0.5) + \frac{0.5}{5}} = 1.66$$

násobek ceny počítače s novým CPU:

$$k = \frac{2}{3} + 5 \cdot \frac{1}{3} = 2.33$$



Amdahlův zákon: celkové zrychlení je rovno

$V_r = 1 / ((1 - f) + f/r)$

Příkon CPU

Mějme procesor Athlon XP 1700+, který pracuje na frekvenci 1.466 GHz a napětí 1.75V.
Jak se změní výkonnost a spotřeba, pokud procesor podtaktujeme na 1GHz (1.4 GHz, 1.266 GHz) a snížíme napájecí napětí na 1.15V (1.6, 1.45V).
Předpokládejme, že procesor v základní konfiguraci má spotřebu 64W.

Pro příkon platí $P \sim C \cdot f \cdot U^2$

$P_{orig,rel} \sim C \cdot 1466 \cdot 1.75^2$ $\qquad$ $P_{orig,w} = 64W$

$P_{novy,rel} \sim C \cdot 1000 \cdot 1.15^2$

$$P_{novy,w} = P_{orig,w} \cdot \frac{P_{novy,rel}}{P_{orig,rel}} = 64 \cdot \frac{1000 \cdot 1.15^2}{1466 \cdot 1.75^2} = 18.56W$$

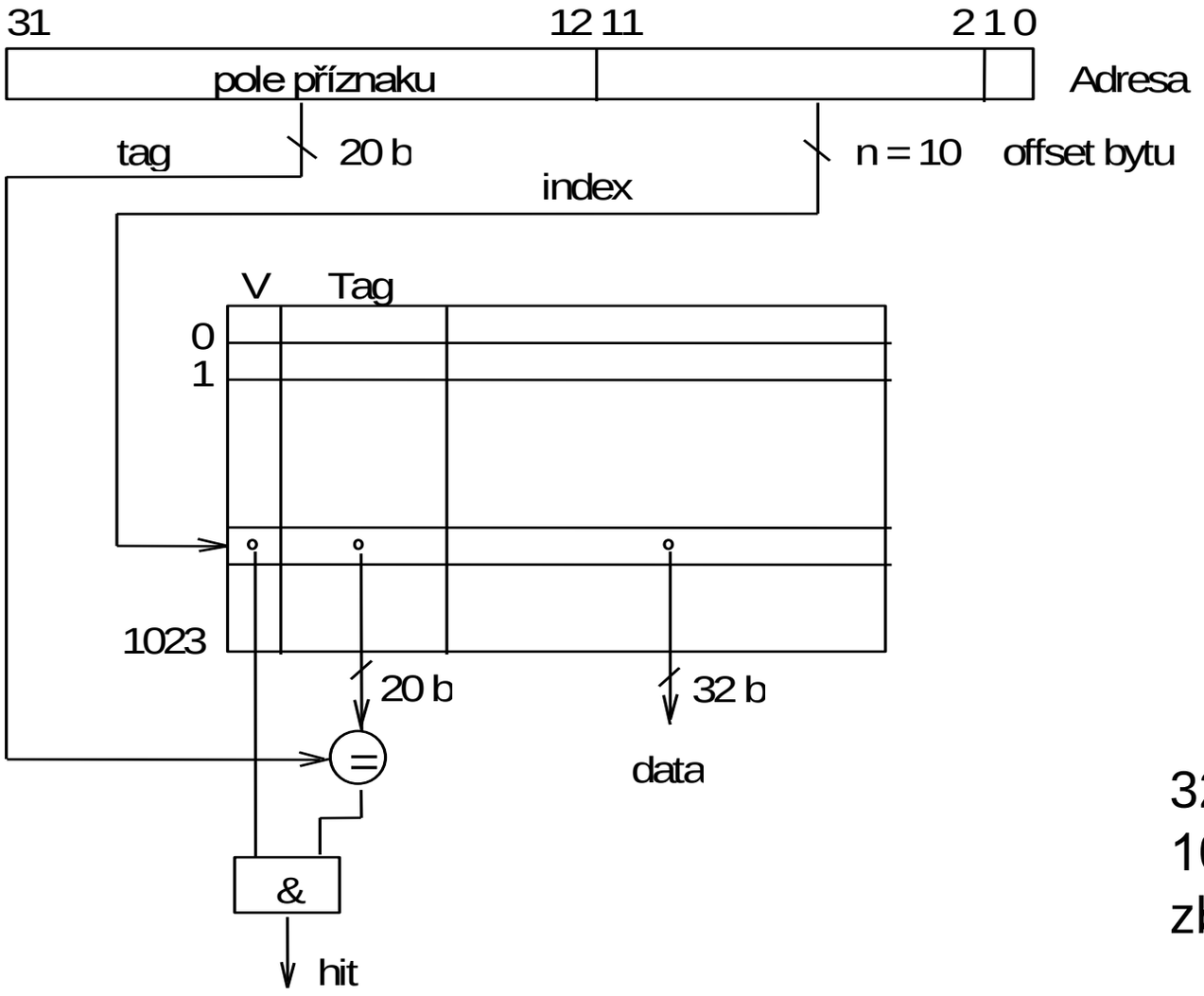
$$f_{rel} = \frac{f_{novy}}{f_{orig}} = \frac{1000}{1466} = 0.68$$

Spotřeba klesla o 71% zatímco výkon CPU pouze o 32%.

RYCHLÁ VYROVNÁVACÍ PAMĚŤ CACHE

CACHE (RVP)

přehled koncepce s 32-bitovou adresou a přímým mapováním



32b dat = 4 B => 2b adresace
1024 položek => 10b adresa
zbytek = 32 – 10 – 2 = 20b tag

Příklad 1

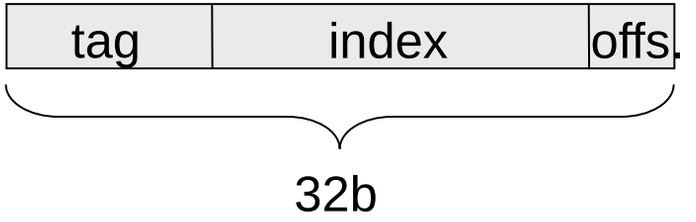
Zadání: Kolik kib paměti bude potřeba pro realizaci jednocestné RVP obsahující 1024 bloků? Uvažujte velikost bloku 64 bitů a 32-bitové adresování.

Řešení

Počet položek … 1024

Počet bitů na jednu položku

Blok dat	… 64 bitů
Uložení tagu	… 32 – $\log_2(1024)$ - $\log_2(64/8)$ = 19 bitů
Valid bit	… 1 bit
Celkem	… 84 bitů



Velikost RVP … 1024 * 84 = 84 kib

Příklad 2

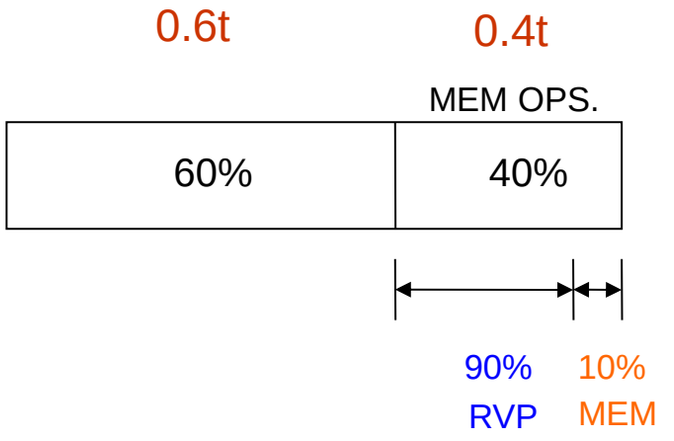
Zadání: Paměťové operace tvoří 40 % času z celkové doby běhu programu. O kolik procent se zkrátí doba běhu programu při použití RVP, když p_{hit} = 0,9 a doba přístupu do RVP je rovna 1/10 doby přístupu do paměti?

$$t_{orig} = t$$

$$t_{RVP} = t_{ops} + t_{mem}$$

$$t_{RVP} = 0.6t + 0.4t\left((1 - p_{hit}) + \frac{p_{hit}}{10}\right)$$

$$t_{RVP} = \big(0.6 + 0.4 \cdot (0.1 + 0.9 \cdot 0.1)\big)t = 0.676t$$

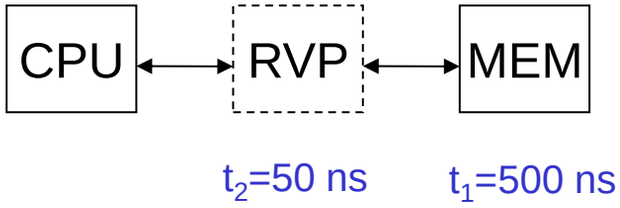


$$1 - \frac{t_{RVP}}{t_{orig}} = 1 - \frac{0.676t}{t} = 0.324 \Rightarrow 32.4\%$$

Příklad 3

Zadání: Mezi procesor a paměť s dobou přístupu 500 ns byla vložena RVP s dobou přístupu 50 ns. Jaká musí být pravděpodobnost zásahu p_{hit} do RVP, aby bylo dosaženo alespoň dvojnásobné urychlení paměťových operací?

$$t_{orig} = t_1 \cdot n = 500 \cdot 10^{-9} \cdot n$$



$$t_{RVP} = \left(t_2 \cdot p_{hit} + (t_2 + t_1) \cdot (1 - p_{hit})\right) \cdot n = (-t_1 \cdot p_{hit} + t_1 + t_2) \cdot n = (550 - 500 \cdot p_{hit}) \cdot 10^{-9} \cdot n$$

$$\frac{t_{orig}}{t_{RVP}} = 2$$

$$\frac{500 \cdot 10^{-9} \cdot n}{(550 - 500 \cdot p_{hit}) \cdot 10^{-9} \cdot n} = 2$$

$$\frac{500}{500 - 500 \cdot p_{hit}} = 2 \qquad \Rightarrow p_{hit} = \frac{1100 - 500}{1000} = \frac{3}{5} = 0.6$$

Cache simulator

- https://mrazekv-students.github.io/bp22_cpu_perina/