کارایی (Performance)

سرفصل

- **تعریف و اندازه‌گیری کارایی پردازنده و کامپیوتر**
  - تعریف کارایی
  - اندازه گیری کارایی و عوامل موثر در کارایی کامپیوتر
  - افزارسنجی (Benchmarking) و نمونه‌های آن
  - دیوار توان

سوالات اساسی

- چرا کارآیی مهم است؟
- چگونه ما می توانیم کارآیی را به دقت تعریف کنیم؟
- چگونه طراحی سخت افزار کارآیی نرم افزار را تحت تأثیر قرار میدهد؟
- چگونه در دنیای واقعی کارآیی اندازه گرفته شود؟
- چرا بعضی از سخت افزارها بهتر از بقیه عمل می کنند؟
- کدام یک از فاکتورهای کارآیی به سخت افزار وابسته اند؟
- ما برای اجرا شدن سریعتر یک برنامه به یک ماشین جدید نیاز داریم یا نیازمند یک سیستم عامل جدید هستیم؟
- مجموعه دستورات یک ماشین چگونه کارآیی را تحت تأثیر قرار می دهند؟

چرا کارآیی مهم است؟

- مقایسه/رتبه‌بندی کامپیوترها
  - کارایی سخت افزار اغلب کلیدی ترین اثر را در کل سیستم متشکل از سخت افزار و نرم افزار است.
  - محصول ارزانتر و/یا بهتر برنده می شود.
    - کامپیوترهای شخصی: رقابت شدید
    - سیستم ها تعبیه شده: بهینه سازی قیمت محصول نهایی
  - برای خریداران مهم ← برای طراحان و تولیدکنندگان مهم است.

- تاثیر عملکرد تغییرات معماری
  - ارزیابی سیستماتیک تنها نشانه ای است که آیا برخی از پیشرفت ها واقعاً یک پیشرفت هستند یا خیر.

نکاتی در باب سوالات اساسی

- اظهار نظر کردن در مورد کارآیی یک سیستم واقعاً کار مشکلی است:
  - سیستم های بزرگ نرم افزاری با جزئیات زیاد + بازهٔ وسیعی از تکنیک های افزایش سرعت که توسط طراحان سخت افزار به کار گرفته می شوند.
- تعیین این که تغییر سخت افزار چقدر برنامه را سریعتر می کند، با توجه به راهنمای مجموعه دستورات، تقریبا غیرممکن است:
  - برای برنامه های کاربردی مختلف، ممکن است مترهای مختلفی مناسب باشد
  - قسمتهای مختلف یک کامپیوتر ممکن است نقش متفاوتی در تعیین سرعت کل سیستم داشته باشند.
- اندازه گیری کارایی صرفا از بیرون مهم نیست بلکه مهم است چه چیزی بر روی آن تاثیر می گذارد.

چطور کارآیی کامپیوتر را تعریف کنیم؟ (۱)

- کارآیی کدام یک از هواپیماهای زیر بیشتراست؟

هواپیما	سرنشین	برد(mi)	سرعت(mph)
Boeing 737-100	101	630	598
Boeing 747	470	4150	610
BAC/Sud Concorde	132	4000	1350
Douglas DC-8-50	146	8720	544

- کدام هواپیما کارایی بهتری دارد؟
  - نیاز به تعریف کارایی دارد.
- طبق معیارهای مختلف:
  - هواپیمار با برد بیشتر: DC-8
  - هواپیمای بزرگتر: boeing 747
  - هواپیمای سریعتر: concorde

- فرض: تعریف کارایی بر حسب سرعت
- دو تعریف مختلف:
  - هواپیمایی که یک مسافر را در کمترین زمان از یک نقطه به نقطه دیگر می برد: کنکورد
  - جابجایی ۴۵۰ مسافر از یک نقطه به نقطه دیگر : بوینگ ۷۴۷

← عملکرد کامپیوتر را به روش های مختلف تعریف کنیم.

چطور کارآیی کامپیوتر را تعریف کنیم؟ (۲)

- می خواهیم سریعترین کامپیوتر را برای کار مورد نظر بخریم.
  - نوع کار اهمیت دارد.

- می خواهیم سریعترین کامپیوتری را طراحی کنیم که مشتریان حاضرند برای آن پول پرداخت کنند.
  - هزینه معیار مهمی است.

تعریف کارآیی

- برای هر کسی چه چیزی مهم است؟

- کاربر سیستم کامپیوتری
  - مدت زمان اجرای برنامه یعنی EndTime – StartTime
  - تحت عنوان زمان پاسخ شناخته می شود.

- ادمین دیتاسنتر
  - حداکثر کردن نرخ اتمام کارها یعنی #jobs/second
  - تحت عنوان گذردهی یا throughput شناخته می شود.

- نیازمند تعریف معیارهای متفاوت برای کارایی بر روی مجموعه برنامه های محک متفاوت هستیم.

زمان پاسخ و گذردهی: مثال

- تغییرات زیر موجب بهبود کدام پارامتر می شود؟ گذردهی، زمان پاسخ، یا هر دو؟
  - CPU سریعتر
    - باعث بهبود هر دو می شود.
  - افزایش تعداد CPU ها
    - گذردهی را قطعا افزایش می دهد.
    - ممکن است به دلیل کاهش طول صف زمان پاسخ را هم بهبود دهد.

کارآیی کامپیوتر چطور تعریف می شود؟

- کارایی بر اساس زمان پاسخ:

$$Performance_x = \frac{1}{ExacuationTime_x}$$

- مقایسه کارایی دو سیستم:

$$Performance_x > Performance_y$$

$$\frac{1}{ExacuationTime_x} > \frac{1}{ExacuationTime_y}$$

$$ExacuationTime_y > ExacuationTime_x$$

کارآیی نسبی

- ماشین x از ماشین y به اندازه ی n برابر سریعتر است اگر:

$$\frac{Performance_x}{Performance_y} = \mathrm{n} = \frac{ExecuationTime_y}{ExecuationTime_x}$$

- ماشین x از ماشین y به اندازه ی n% سریعتر است اگر:

$$\frac{Performance_x}{Performance_y} = \frac{ExecuationTime_y}{ExecuationTime_x} = 1 + n/100$$

- مثال: ExacuationTime(A) = 10s, ExacuationT ime(B) = 15s ؟
  - A یک و نیم برابر از B سریعتر است.
  - A ۵۰ درصد از B سریعتر است.

سرفصل

- **تعریف و اندازه‌گیری کارایی پردازنده و کامپیوتر**
  - تعریف کارایی (معکوس زمان اجرا)
  - اندازه گیری کارایی و عوامل موثر در کارایی کامپیوتر
  - افزارسنجی (Benchmarking) و نمونه‌های آن
  - دیوار توان

کارآیی: از دید کاربر (۱)

- **کل زمان اجرا (تکمیل کار)**
  - Wall-clock time, response time, elapsed time

- شامل انتظار برای عملیات I/O، سربار سیستم عامل و غیره است.
  - شامل زمان اشتراک منابع (CPU) با سایر کاربران است.
- کارایی کلی سیستم را منعکس می کند.

کارآیی: از دید کاربر

- **زمان پردازنده**
  - CPU execution time, CPU time
  - زمانی که برنامه واقعاً در حال اجرا بوده است.
    - انتظار برای عملیات I/O را شامل نمی شود
    - شامل زمان اجرا نشدن برنامه مثل سربار سیستم عامل نمی شود
  - عملکرد پردازنده را منعکس می کند.

  - دو بخش CPU time:
    - زمانی از cpu که صرف اجرای خود برنامه می شود (cpu time user)
    - زمانی از cpu که صرف اجرای سیستم عامل در ارتباط با این برنامه می شود (system cpu time)

کارآیی: از دید طراح

- سرعت اجرای دستورات
  - Clock rate
  - Clock cycle length

$$CPU\ execution\ time = CPU\ clock\ cycle\ \times CPU\ clock\ cycle\ time$$

$$CPU\ execution\ time = \frac{CPU\ clock\ cycle}{CPU\ clock\ rate}$$

مثال: کارآیی از دید طراح (۱)

- برنامه ما در ۱۰ ثانیه روی کامپیوتر A که دارای ساعت ۲ گیگاهرتز است اجرا می شود. سعی می کنیم به یک طراح کامپیوتر کمک کنیم تا یک کامپیوتر B بسازد که این برنامه را در ۶ ثانیه اجرا می کند.
- طراح تشخیص داده است که افزایش قابل توجهی در نرخ کلاک امکان پذیر است، اما این افزایش بر بقیه طراحی CPU تأثیر می گذارد و باعث می شود رایانه B به ۱.۲ برابر چرخه ساعت بیشترنسبت به رایانه A برای این برنامه به چرخه ساعت نیاز داشته باشد.
- چه نرخ ساعتی را باید به طراح بگوییم که هدف گذاری کند؟

مثال: کارآیی از دید طراح (۲)

$$\text{CPU time}_A = \frac{\text{CPU clock cycles}_A}{\text{Clock rate}_A}$$

$$10 \text{ seconds} = \frac{\text{CPU clock cycles}_A}{2 \times 10^9 \frac{\text{cycles}}{\text{second}}}$$

$$\text{CPU clock cycles}_A = 10 \text{ seconds} \times 2 \times 10^9 \frac{\text{cycles}}{\text{second}} = 20 \times 10^9 \text{ cycles}$$

$$\text{CPU time}_B = \frac{1.2 \times \text{CPU clock cycles}_A}{\text{Clock rate}_B}$$

$$6 \text{ seconds} = \frac{1.2 \times 20 \times 10^9 \text{ cycles}}{\text{Clock rate}_B}$$

$$\text{Clock rate}_B = \frac{1.2 \times 20 \times 10^9 \text{ cycles}}{6 \text{ seconds}} = \frac{0.2 \times 20 \times 10^9 \text{ cycles}}{\text{second}} = \frac{4 \times 10^9 \text{ cycles}}{\text{second}} = 4 \text{ GHz}$$

کارآیی: از دید کامپایلر

- هر برنامه شامل تعدادی دستورالعمل است.
  - سخت افزار دستورالعملها را می شناسد نه برنامه ها را.

- میانگین تعداد سیکل به ازاء هر دستور
- Clock cycles per instruction (CPI)
- مخصوص یک برنامه خاص یا بخشی از آن
- امکان مقایسه پیاده سازی های مختلف از یک معماری را فراهم می کند
  - با توجه به تعداد ثابتی از دستورالعمل ها

$$CPU\ Clock\ cycles = CPI \times Number\ of\ Instructions$$

معادله کلاسیک کارایی پردازنده (۱)

- به تعداد دستورالعمل ها، CPI و طول سیکل ساعت مربوط می شود.
- ۳ فاکتور مختلف موثر بر عملکرد
  - امکان مقایسه پیاده سازی های مختلف را فراهم می کند
  - امکان ارزیابی معماری های جایگزین را فراهم می کند

$$CPU\ time = CPI \times Number\ of\ Instructions \times CPU\ clock\ cycle\ time$$

$$CPU\ time = \frac{CPI \times Number\ of\ Instructions}{CPU\ clock\ rate}$$

Components of performance	Units of measure
CPU execution time for a program	Seconds for the program
Instruction count	Instructions executed for the program
Clock cycles per instruction (CPI)	Average number of clock cycles per instruction
Clock cycle time	Seconds per clock cycle

معادله کلاسیک کارایی پردازنده (۲)

$$time = \frac{Seconds}{Program} = \frac{Instruction}{Program} \times \frac{Clock\ cycles}{Instruction} \times \frac{Seconds}{Clock\ cycle}$$

- مثال:
  - 500MHz P-III در هر ثانیه 500M دستور اجرا می کند و هر سیکل برابر 2ns طول می کشد.
  - 2GHz P-4 در هر ثانیه 2G دستور اجرا می کند و هر سیکل برابر 0.5ns طول می کشد.

مثال ۱: معادله کلاسیک کارایی (۱)

- فرض کنید ما دو پیاده سازی از یک مجموعه دستورات واحد در اختیار داشته باشیم. ماشین A برای یک برنامه دارای پریود کلاک ۱ نانوثانیه بوده و CPI آن ۲ می باشد و ماشین B دارای پریود کلاک ۲ نانوثانیه بوده و CPI آن ۱/۲ می باشد. برای این برنامه کدام ماشین سریعتر است و چقدر؟
  - تعداد دستوراتی که هر کدام از ماشینها برای این برنامه اجرا می کنند یکی است ← تعداد دستورات= I

تعداد کلاکهای لازم برای اجرای برنامه بر روی ماشین A = I × 2

تعداد کلاکهای لازم برای اجرای برنامه بر روی ماشین B = I × 1.2

دوره تناوب کلاک × تعداد کلاکهای لازم برای اجرای برنامه = زمان اجرای برنامه بر روی ماشین A

= I × 2.0 × 1 ns = 2 × I n

زمان اجرای برنامه بر روی ماشین B = I × 1.2 × 2 ns = 2.4 × I

$$\frac{Performance_A}{Performance_B} = n = \frac{ExecuationTime_B}{ExecuationTime_A} = 2.4I/2I = 1.2$$

مثال ۲: معادله کلاسیک کارایی (۱)

- یک طراح کامپایلر می خواهد بین دو قطعه کد برای یک ماشین مشخص یکی را انتخاب نماید.
- بر اساس پیاده سازی سخت افزار، سه کلاس مختلف از دستورات وجود دارد: کلاس A، کلاس B و کلاس C که به ترتیب برای اجرا شدن نیاز به ۲ ،۱و ۳ کلاک دارند.
- قطعه کد اول دارای ۵ دستور است: ۲ مورد از کلاس A، ۱ مورد از کلاس Bو ۲ مورد از کلاس C و قطعه کد دوم دارای ۶ دستور است: ۴مورد از کلاس A، ۱ مورد از کلاس B و ۱مورد از کلاس C
  - کدام قطعه کد سریعتر اجرا میشود و چقدر؟
  - CPI را برای هر قطعه کد حساب کنید؟

مثال ۲: معادله کلاسیک کارایی (۲)

CPU clock time = $\sum_{i=1}^{n} CPI_i \times C_i$

- CPIi تعداد کلاکهای لازم برای اجرای دستورات از نوع کلاس I
- Ci تعداد دستورات کلاس I

$$CPU\ clock\ time_1 = 2 * 1 + 1 * 2 + 2 * 3 = 10$$
$$CPU\ clock\ time_2 = 4 * 1 + 1 * 2 + 1 * 3 = 9$$

$$\text{CPI} = \frac{CPU\ clock\ time}{NUmber\ o\ fInstruction}$$

$$CPI_1 = \frac{10}{5} = 2$$

$$CPI_2 = \frac{9}{6} = 1.5$$

مثال ۳: معادله کلاسیک کارایی (۲)

- دو پروسسور زیر را که دارای ISA یکسان یا سازگار می باشند از نظر کارایی با هم مقایسه کنید؟
  - یک پردازنده ۸۰۰ مگاهرتز AMD Duron که CPI آن ۱/۲ است.
  - یک پردازنده ۱ گیگا هرتز پنتیوم III که CPI آن ۱/۵ است.

- ISA یکسان ← تعداد دستورات یکسان برای یک برنامه فرضی

$$CPU\ time = \frac{CPI \times Number\ of\ Instructions}{CPU\ clock\ rate}$$

$$AMD\ CPU\ time = \frac{1.2 \times I}{800\ MHz} = 1.5\ \text{ns} \times I$$

$$Pentium\ III\ CPU\ time = \frac{1.5 \times I}{1\ GHz} = 1.5\ \text{ns} \times I$$

درک کارایی برنامه

Component	Affects what?	Affects how?
Algorithm	Instruction count, CPI	Number and kind of source program statements and operations, data types (integer vs. floating point)
Programming Language	Instruction count, CPI	Kind of source program statements, abstractions used to express the algorithm.
Compiler	Instruction count, CPI	How program statements are translated to machine code, choice and layout of instructions.
Instruction set architecture	Instruction count, CPI, Clock rate	Instructions available to compiler, cost in cycles for each instruction, overall clock rate.

تله: انتظارات غیرواقعی (۱)

- انتظار بهبود یکی از جنبه های کامپیوتر برای افزایش کارایی کلی به میزانی متناسب با اندازه بهبود.
  - کل زمان اجرا: ۱۰۰ ثانیه
    - از این تعداد، عملیات ضرب: ۸۰ ثانیه
  - چقدر به بهبود ضرب نیاز داریم تا برنامه ۵ برابر سریعتر اجرا شود؟

تله: انتظارات غیرواقعی (۲)

$$Execution_{fast} = \frac{Execution_{slow}}{5}$$

$$Execution_{slow} = 80 + 20$$

$$Execution_{fast} = \frac{80}{n} + 20$$

$$\frac{80}{n} + 20 = \frac{80+20}{5}$$

$$\frac{80}{n} + 20 = 20$$

$$\frac{80}{n} = 0$$

قانون آمدال (۱)

- Gene Amdahl (1922)
- نسخه های مختلف
- قانونی است که می توان به کمک آن ارزش کارهایی که قرار است انجام شوند را از قبل پیش بینی نمود. این کارها عمدتاً مربوط به بهبودهایی است که در سخت افزار پردازنده ها داده می شود
  - مقدار بهبودهایی که انجام می شود به میزان مؤثر بودن آنها محدود می شود
  - فرض کنید کسری از سیستم را (یعنی f) را به اندازه ی s برابر بهبود می دهیم.

$$Speedup = \frac{[(1-f)+f] \times oldtime}{[(1-f) \times oldtime] + \frac{f}{s} \times oldtime} = \frac{1}{1-f+\frac{f}{s}}$$

قانون آمدال (۲)

- *انگیزه:* بهبود حالتهای رایج
- تاثیر عملی
  - موارد رایج (پراستفاده) را سریع کنید
  - بهینه سازی بیشترین تأثیر را روی مورد رایج دارد
    - حالت رایج اغلب بسیار ساده تر از موارد خاص است و بنابراین بهینه سازی آن آسان تر است
  - حتی بهینه‌سازی انبوه موارد خاص اغلب در مقایسه با بهینه‌سازی عادی موارد معمول، سود بسیار کمی دارد.

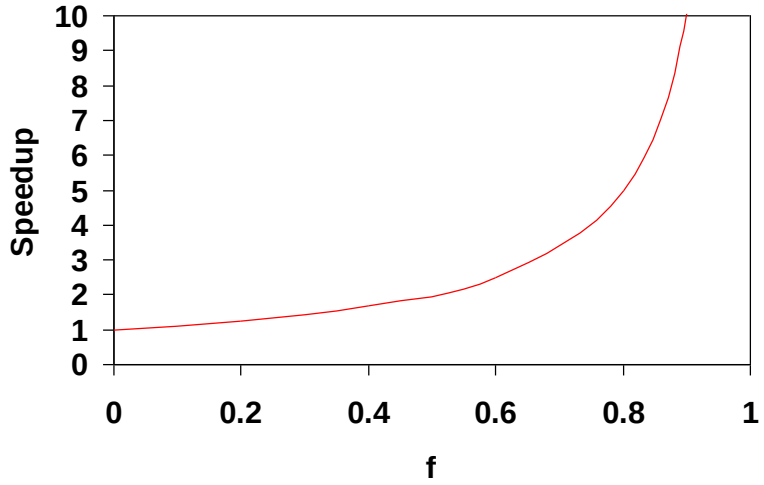
قانون آمدال (۳)

- رییس شما از شما می خواهد یکی از بهبودهای زیر را انجام دهید.
  - ALU را ۱۰٪ بهبود دهید. ALU در ۹۵٪ اوقات مورد استفاده قرار می گیرد.
  - حافظه خط لوله را ۱۰ برابر بهبود دهید. حافظه در ۵٪ اوقات مورد استفاده قرار می گیرد.

f	s	Speedup
95%	1.10	1.094
5%	10	1.047
5%	∞	1.052

قانون آمدال: حد

$$\lim_{s\to\infty} \frac{1}{1-f+\frac{f}{s}} = \frac{1}{1-f}$$



- اگر (1-f) ناچیز نباشد، بهبود محدود است.

- مثال: افزایش تعداد پردازنده ها
  - فرض کنید از ۱۰۲۴ پردازنده ی موازی برای بهبود یک برنامه استفاده کنیم.
    - قسمت موازی برنامه ۱۰۲۴ برابر بهتر می شود.
  - قسمت سریال برنامه بهبود نهایی را به 1-f محدود می کند.
    - مثلاً اگر قسمت سریال ۱۰٪ باشد، حداکثر می توان برنامه را ۱۰ برابر سریعتر کرد.

تله: معیارهای اشتباه کارایی (۱)

- استفاده از زیرمجموعه ای از معادله کارایی به عنوان معیار کارایی
  - استفاده از یک فاکتور تقریباً همیشه اشتباه است.
  - استفاده از دو عامل ممکن است در زمینه محدود معتبر باشد
    - به سادگی اشتباه استفاده می شود: وابستگی بین عوامل
  - معیارهای دیگر که سایر عوامل شناخته شده را پوشش می دهند.

تله: معیارهای اشتباه کارایی (۲)

- MIPS (Million Instructions Per Second) $$MIPS = \frac{Instruction\ count}{10^6 \times Executiontime}$$
- نرخ اجرای دستورالعمل (تعداد بیشتر ← رایانه سریعتر).
- اشکالات:
  - نادیده گرفتن قابلیت های دستورالعمل، زمان اجرای هر دستورالعمل، تعداد متفاوت دستورالعمل برای ISAهای مختلف
    - مقایسه کامپیوترها با ISAهای مختلف غیرممکن است
  - به ترکیب دستورات یک برنامه خاص بستگی دارد (یک مقدار برای نشان دادن عملکرد یک کامپیوتر)
  - اگر یک برنامه جدید دستورالعمل های بیشتری را اجرا کند اما هر دستورالعمل سریعتر باشد، MIPSمی تواند مستقل از عملکرد متفاوت باشد!
- چه زمان استفاده از MIPS اشکالی ندارد؟
  - کامپایلر و ISA یکسان باشند.
  - مثل اجرای یک کد باینری روی i3 و i5 ، زیرا در این حالت ترم Instr/program ثابت است و قابل صرفه نظر کردن است.

کارایی پردازنده (۱)

- عملکرد در هنگام اجرای یک برنامه خاص
  - به تعداد دستورالعمل ها، میانگین تعداد چرخه ها در هر دستورالعمل (CPI)، طول چرخه ساعت (یا نرخ ساعت) بستگی دارد.
    - هیچ عامل واحدی نمی تواند عملکرد را به طور کامل بیان کند.
      - کاهش تعداد دستورالعمل ها ← معماری با فرکانس ساعت کمتر یا CPI بالاتر
      - CPI به ترکیب دستورات (فرکانس و نوع دستورالعمل های اجرا شده) یک برنامه مشخص بستگی داردکد با کمترین تعداد دستورالعمل لزوما سریعترین نیست.

کارایی پردازنده (۲)

- عملکرد در هنگام اجرای یک برنامه خاص
  - تنها معیار کامل و قابل اعتماد، زمان پردازنده است
    - چیزی در مورد زمان پردازنده برای برنامه های دیگر نمی گوید.

سرفصل

- **تعریف و اندازه‌گیری کارایی پردازنده و کامپیوتر**
  - تعریف کارایی (معکوس زمان اجرا)
  - اندازه گیری کارایی
  - عوامل موثر در کارایی کامپیوتر
  - افزارسنجی (Benchmarking) و نمونه‌های آن
  - دیوار توان

ارزیابی پردازنده (۱)

- مقایسه عملکرد کامپیوترهای مختلف
  - آسان برای یک برنامه خاص (زمان اجرای پردازنده)
  - مقایسه اجزای مجزا (نرخ ساعت، CPI، تعداد دستورالعمل ها) نشان دهنده برنامه های دیگر نیست.
  - چگونه می توان کارایی را با توجه به مجموعه ای از برنامه ها تقریب زد؟ زمان اجرای کدام برنامه مبنای مقایسه است؟
    - حالت بهینه: فرض کنید شما همیشه از مجموعه ی مشخص و یکسانی از برنامه ها استفاده می کنید.
    - می توانید برنامه هایتان را روی هر دو ماشین اجرا کنید.

ارزیابی پردازنده (۲)

- **بار کار (Workload)**
  - مجموعه ای از برنامه ها و وظایف که حجم کاری کاربر را به تصویر می کشد یا توسط کاربر اجرا می شود.
  - مشکل تعریف (برای دامنه خاص)
  - مشکل خودکارسازی (اجراهای تکراری)
- **محک (Benchmark)**
  - برنامه ای که به طور خاص برای اندازه گیری کارایی ساخته شده است.
  - باعث صرفه جویی در وقت و هزینه می شود.
  - مجموعه ای از محک ها
    - نمایش متناسب آماری بار کاری معمولی (typical workload)
    - امیدواریم که نتایج بنچمارک نشان دهنده میزان عملکرد یک رایانه با بار کاری کاربر باشد.

مخاطرات Benchmark

- ممکن است Benchmark نمایانگر خوبی نباشد.
  - مثلاً اگر عمده ی کار شما I/O باشد، SPEC استفاده ای ندارد.
  - benchmarkهایی که اندازه آنها کوچک است و یا کارآیی آنها به قسمت کوچکی از کد برنامه بستگی دارد.

- Benchmark قدیمی است.
  - سریع قدیمی می شود و فرصت سوء استفاده برای سازندگان سخت افزار پدید می آید تا با بهینه کردن کامپایلر، سخت افزار و نرم افزار نسبت به benchmark ها پردازنده خود را بهتر جلوه نمایند.
  - باید مرتبا به روز شود.

گزارش کارایی Benchmark

Hardware	
Model number	Powerstation 550
CPU	41.67 MHz POWER 4164
FPU	Integrated
Number of CPUs	1
Cache size per CPU	64K data / 8K instruction
Memory	64 MB
Disk subsystem	2 400-MB SCSI
Network interface	NA
Software	
OS type and rev	AIX v3.1.5
Compiler rev	AIX XL C/6000 Ver. 1.1.5 AIX XL Fortran Ver. 2.2
Other software	None
File system type	AIX
Firmware level	NA
System	
Tuning parameters	None
Background load	None
System state	Multiuser (single-user-login)

- توضیحاتی که در گزارش آورده می شود باید طوری باشد که اندازه گیری قابل تکرار باشد.
- گزارش باید شامل یک لیست تمام مواردی باشد که ممکن است یک فرد دیگر برای تکرار آزمایش ها به آنها نیاز داشته باشد.
  - نسخه سیستم عامل، کامپایلرها، ورودیها و همچنین نحوه پیکره بندی ماشین

محک SPEC (Standard Performance Evaluation Corporation)

- سرمایه گذاری توسط نهادهای تجاری و غیر تجاری
  - تولید کنندگان پردازنده و کامپیوتر
  - تولید کنندگان کامپایلر، سیستم عامل
  - موسسات تحقیقاتی
- هدف: تعریف یک مجموعه استاندارد از معیارها برای امکان مقایسه عملکرد سیستم های کامپیوتری
  - معیارهای مختلف برای بارهای کاری مختلف
  - عمدتاً بر کارایی CPU تمرکز دارد
  - در حال حاضر توان عملکردی CPU، عملکرد GPU، کامپایلرها، پایگاه های داده، سیستم های پست الکترونیکی، و ...
- SPEC89, SPEC92, SPEC95, SPEC2000 و SPEC2006

SPEC CPU2006

- کارایی پردازنده
  - CINT2006 (محاسبات صحیح)
    - ۱۲ برنامه محک (کامپایلر C، الگوریتم شطرنج، شبیه سازی کامپیوتر کوانتومی و غیره)
  - CFP2006 (محاسبات ممیز شناور)
    - ۱۷ برنامه محک (عناصر محدود، دینامیک مولکولی و غیره)
  - SPECratio
    - نسبت مرجع به زمان اجرای بنچمارک اندازه گیری شده به صورت نرمال شده
    - امتیاز خلاصه (تک عدد): میانگین هندسی

$$\sqrt[n]{\prod_{i=1}^{n} SPECratio_i}$$

PerforSPEC CPU 2006 on AMD Opteron X4

Description	Name	Instruction Count × 10^9	CPI	Clock cycle time (seconds × 10^9)	Execution Time (seconds)	Reference Time (seconds)	SPECratio
Interpreted string processing	perl	2,118	0.75	0.4	637	9,770	15.3
Block-sorting compression	bzip2	2,389	0.85	0.4	817	9,650	11.8
GNU C compiler	gcc	1,050	1.72	0.4	724	8,050	11.1
Combinatorial optimization	mcf	336	10.00	0.4	1,345	9,120	6.8
Go game (AI)	go	1,658	1.09	0.4	721	10,490	14.6
Search gene sequence	hmmer	2,783	0.80	0.4	890	9,330	10.5
Chess game (AI)	sjeng	2,176	0.96	0.4	837	12,100	14.5
Quantum computer simulation	libquantum	1,623	1.61	0.4	1,047	20,720	19.8
Video compression	h264avc	3,102	0.80	0.4	993	22,130	22.3
Discrete event simulation library	omnetpp	587	2.94	0.4	690	6,250	9.1
Games/path finding	astar	1,082	1.79	0.4	773	7,020	9.1
XML parsing	xalancbmk	1,058	2.70	0.4	1,143	6,900	6.0
Geometric Mean							11.7

PerforSPEC CPU 2006 on Intel Core i7 920

Description	Name	Instruction Count x 10^9	CPI	Clock cycle time (seconds x 10^{-9})	Execution Time (seconds)	Reference Time (seconds)	SPECratio
Interpreted string processing	perl	2252	0.60	0.376	508	9770	19.2
Block-sorting compression	bzip2	2390	0.70	0.376	629	9650	15.4
GNU C compiler	gcc	794	1.20	0.376	358	8050	22.5
Combinatorial optimization	mcf	221	2.66	0.376	221	9120	41.2
Go game (AI)	go	1274	1.10	0.376	527	10490	19.9
Search gene sequence	hmmer	2616	0.60	0.376	590	9330	15.8
Chess game (AI)	sjeng	1948	0.80	0.376	586	12100	20.7
Quantum computer simulation	libquantum	659	0.44	0.376	109	20720	190.0
Video compression	h264avc	3793	0.50	0.376	713	22130	31.0
Discrete event simulation library	omnetpp	367	2.10	0.376	290	6250	21.5
Games/path finding	astar	1250	1.00	0.376	470	7020	14.9
XML parsing	xalancbmk	1045	0.70	0.376	275	6900	25.1
Geometric mean	–	–	–	–	–	–	25.7

SPECpower_ssj2008

با توجه به اهمیت روزافزون انرژی و توان، SPEC یک بنچمارک برای اندازه گیری توان اضافه کرد.

Target Load %	Performance (ssj_ops)	Average Power (watts)
100%	865,618	258
90%	786,688	242
80%	698,051	224
70%	607,826	204
60%	521,391	185
50%	436,757	170
40%	345,919	157
30%	262,071	146
20%	176,061	135
10%	86,784	121
0%	0	80
Overall Sum	4,787,166	1922
∑ssj_ops / ∑power =		2490

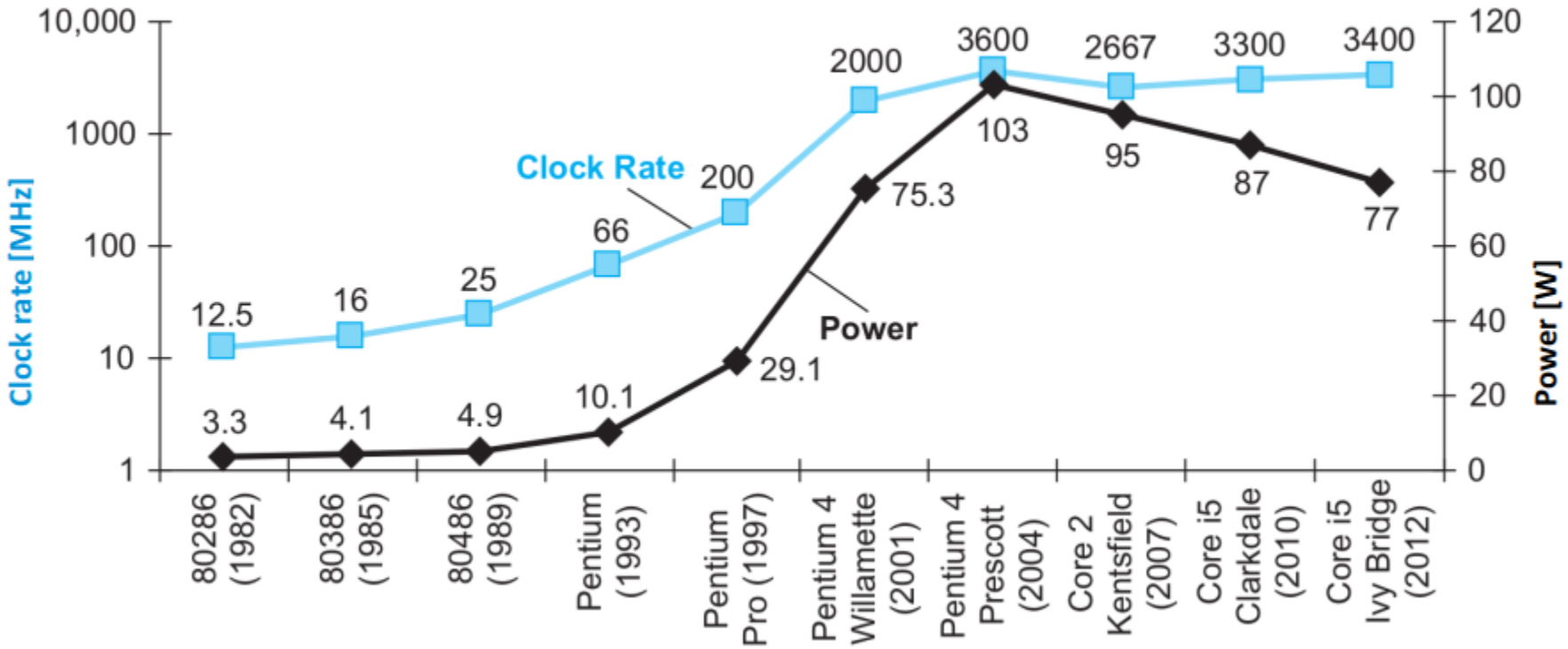
FIGURE 1.19 SPECpower_ssj2008 running on a dual socket 2.66 GHz Intel Xeon X5650 with 16 GB of DRAM and one 100 GB SSD disk.

سرفصل

- **تعریف و اندازه‌گیری کارایی پردازنده و کامپیوتر**
  - تعریف کارایی (معکوس زمان اجرا)
  - اندازه گیری کارایی
  - عوامل موثر در کارایی کامپیوتر
  - افزارسنجی (Benchmarking) و نمونه‌های آن
  - دیوار توان

پایان دوره طلایی

دیوار توان (۱)


Clock rate [MHz]
Power [W]
Clock Rate
Power
10,000
1000
100
10
1
120
100
80
60
40
20
0
12.5
16
25
66
200
2000
3600
2667
3300
3400
3.3
4.1
4.9
10.1
29.1
75.3
103
95
87
77
80286 (1982)
80386 (1985)
80486 (1989)
Pentium (1993)
Pentium Pro (1997)
Pentium 4 Willamette (2001)
Pentium 4 Prescott (2004)
Core 2 Kentsfield (2007)
Core i5 Clarkdale (2010)
Core i5 Ivy Bridge (2012)

دیوار توان (۲)

- Complementary Metal Oxide Semiconductor (CMOS)
  - تکنولوژی غالب برای مدارهای مجتمع
    - مصرف استاتیک بسیار کم
    - مصرف توان دینامیک
      - بار خازنی (رساناها، ترانزیستورها، بار خروجی)
      - ولتاژ کاری (بر سرعت سوئیچینگ تأثیر می گذارد)
      - فرکانس سوئیچینگ (تابعی از نرخ کلاک)

$$Power \approx \frac{1}{2} \times Capacitive\ load \times Voltage^2 \times Frequency\ switched$$

دیوار توان (۳)

- **تاثیر دنیای واقعی**
  - در ۲۰ سال گذشته
  - رشد نرخ ساعت با ضریب ۱۰۰۰
  - رشد توان (فقط) با ضریب ۳۰، چگونه؟
  - کاهش ولتاژ از ۵ ولت به ۱ ولت
  - ۱۵ درصد کاهش با هر نسل

- **مثال**
  - فنآوری جدید منجر به ۸۵ درصد بار خازنی فناوری قدیمی می شود. همچنین می توان ولتاژ کاری و فرکانس سوئیچینگ را تا ۱۵ درصد کاهش داد تا در مصرف برق صرفه جویی شود.
  - در مقایسه با یک پردازنده مبتنی بر فناوری قبلی، یک پردازنده جدید تنها ۵۲ درصد از انرژی را مصرف می کند.

دیوار توان (۴)

- کاهش بیشتر ولتاژ دشوار/غیر ممکن است
  - ترانزیستورها را بیش از حد نشتی می کند
  - ۴۰ درصد مصرف برق در تراشه های سرور به دلیل نشتی است
  - نسبت سیگنال/نویز پایین
    - تشخیص یک ها از صفر به طور قابل اعتماد دشوار است.

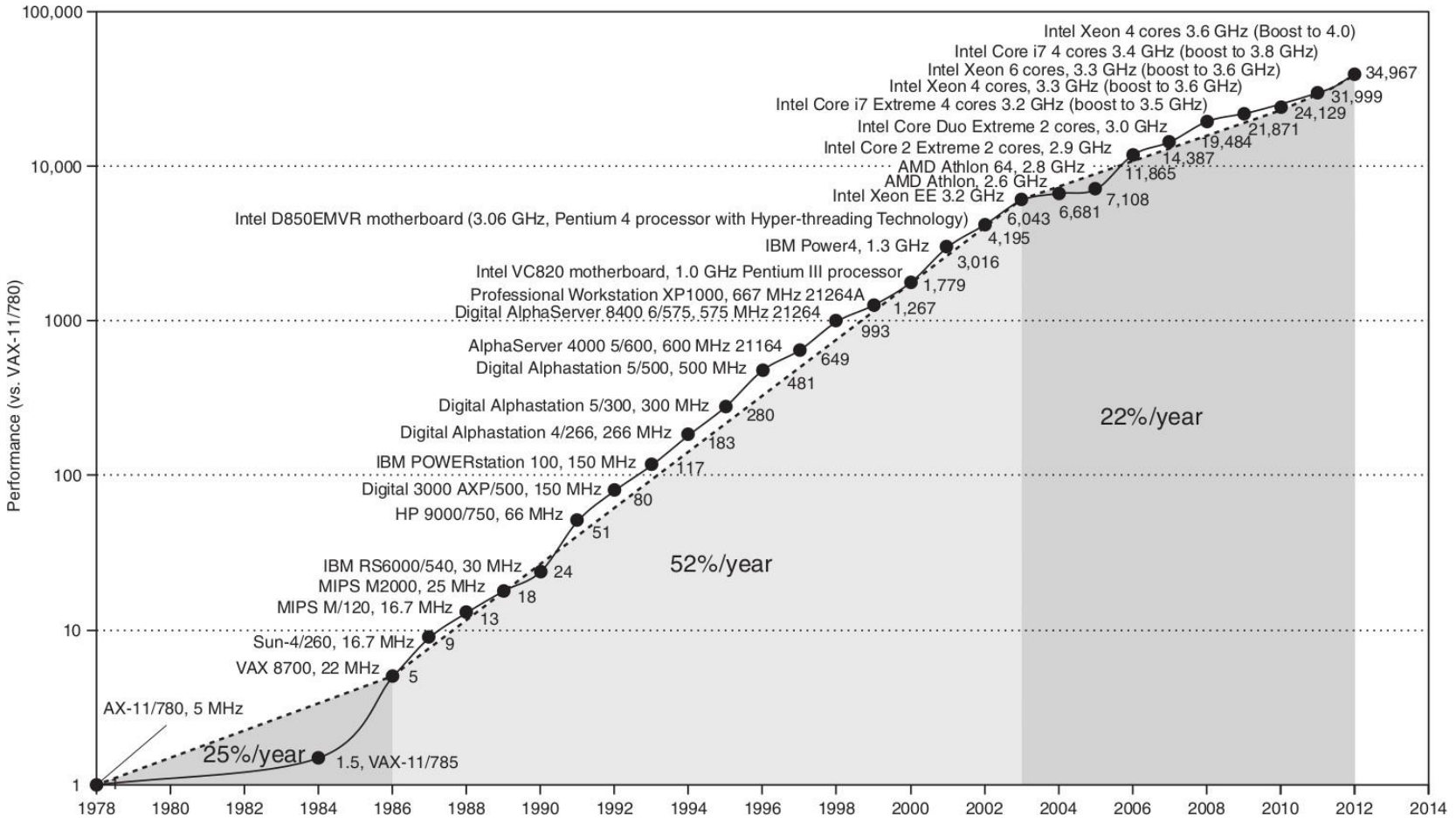
- خنک کننده را نمی توان به راحتی بهبود بخشید
  - انرژی از ناحیه نسبتاً کوچکی از تراشه تلف می شود.
  - بخش هایی از تراشه که در چرخه ساعت استفاده نمی شوند را می توان خاموش کرد.
  - تکنیک‌های خنک‌کننده آب (و دیگر) بسیار پیچیده/گران‌قیمت است.
    - حتی گزینه ای برای دستگاه های تلفن همراه نیست

دیوار توان (۵)

- راه حل جدیدی برای بهبود کارایی مورد نیاز است
  - تغییر چشمگیر در طراحی ریزپردازنده

تغییر از Uniprocessors به Multiprocessors

رشد کارایی تک پردازنده



سیستم های چند پردازنده ای

- بعدا
  - چند پردازنده فیزیکی (چند پردازنده)
  - کجا: ابر رایانه ها، سرورهای پیشرفته
  - استفاده در رایانه های شخصی و جاسازی شده نادر است
- در حال حاضر
  - چندین هسته پردازنده در یک بسته ریزپردازنده واحد
  - در دنیای قانون Post-Moore، کوچک شدن ترانزیستورها دشوار/گران است، اما هنوز هم می‌توانیم مقدار بیشتری از آن را روی یک تراشه (بزرگتر) قرار دهیم
  - کجا: همه جا

سیستم های چند هسته ای

- **تاثیر بر کارایی**
  - افزایش توان عملیاتی
    - پردازش درخواست های بیشتر به صورت موازی
  - نرخ کلاک و CPI ثابت می مانند
    - عملکرد الگوریتم های ترتیبی ثابت می ماند.

- **تاثیر بر برنامه نویسان**
  - فناوری برنامه ها را سریع تر نمی کند (بیش از این)
  - برنامه ها باید از مزایای چندین هسته بهره ببرند
    - API های بهتر مورد نیاز است (فریم ورک های اجرایی، مجموعه های موازی، ...)
  - با افزایش تعداد هسته ها، برنامه ها باید بهبود یابند
    - افزایش تعداد هسته ها از ۴ به ۳۲، برنامه موازی را ۸ برابر سریعتر نمی کند.

چرا یک معامله مهم است؟

- تغییر اساسی در رابط HW/SW
  - موازی بودن همیشه مهم بود، اما قبلا پنهان بود.
    - موازی سازی در سطح دستورالعمل، خط لوله، و تکنیک های دیگر.
    - برنامه نویس و کامپایلر به طور یکسان کدهای ترتیبی تولید کردند.
  - حالا موازی بودن باید صریح باشد!
- معماری های موازی برای ۴۰ سال شناخته شده بودند
  - ..... اما هر که بر موازی سازی صریح تکیه کرد شکست خورد!
    - برنامه نویسان هرگز پارادایم جدید را نپذیرفتند.
  - اکنون کل صنعت فناوری اطلاعات روی برنامه نویسان شرط بندی می کند تا به موازی سازی صریح روی بیاورند.

چرا برنامه نویسی موازی دشوار است؟ (۱)

- برنامه نویسی با تمرکز بر کارایی
  - سختی برنامه نویسی را افزایش می دهد
    - برنامه نه تنها باید صحیح باشد، بلکه باید سریع نیز باشد
    - اگر به عملکرد نیاز ندارید، فقط یک برنامه ترتیبی بنویسید.
  - مردم به "sequentially" در یک "single thread" فکر می کنند

- مشکل: تقسیم کار به طور مساوی بین پردازنده ها
  - اطمینان حاصل کنید که سربار برنامه ریزی و هماهنگی کار مزیت کارایی را از بین نمی برد

چرا برنامه نویسی موازی دشوار است؟ (۲)

- مثالی از دنیای واقعی
  - فرض: ۱ روزنامه نگار در ۲ ساعت ۱ مقاله می نویسد.
  - سوال: آیا می توانیم ۸ روزنامه نگار را وادار کنیم که در ۱۵ دقیقه یک مقاله بنویسند؟
  - مشکلات واقعی
    - زمانبندی
      - هر فردی چه چیزی را می نویسد
    - تعادل بار
      - هیچ روزنامه نگاری بیکار نیست
    - سربار ارتباط و هماهنگ سازی
      - چگونه مقاله نهایی را کنار هم بگذاریم؟

خلاصه و نکات پایانی

- دو روش معتبر برای اندازه گیری کارآیی وجود دارد: محاسبه زمان اجرا و محاسبه throughput
- روش قابل اطمینان برای اندازه گیری کارآیی: استفاده از زمان اجرای برنامه های واقعی بر روی پردازنده ها

دوره تناوب کلاک × CPI × تعداد دستورات = زمان اجرا

- هیچ کدام از این فاکتورها به تنهایی نمی توانند نشان دهنده کارآیی باشند.
- بنچمارک ها برنامه هایی هستند که برای اندازه گیری کارآیی انتخاب می شوند:
  - باید به اندازه کافی بزرگ باشند که اجرای آنها توسط کامپایلر و یا سخت افزار بهینه سازی نگردد
  - تا حد ممکن از برنامه های واقعی انتخاب شوند.
- مهمترین مجموعه benchmarkهای شناخته شده مجموعه SPEC است.

نکات پایانی

- قانون امدال: از یک بهبود در طرح چقدر می توانیم انتظار بهبود کارآیی را داشته باشیم.
  - میزان بهبود کارآیی به میزان مؤثر بودن تغییر داده شده بستگی دارد.
- گزارش تهیه شده برای کارآیی باید طوری باشد که توسط دیگران قابل تکرار باشد.
- هنر طراحی کامپیوتر فقط اعمال کردن عدد در فرمول کارآیی نیست بلکه مشخص کردن این موضوع است که طراحی های مختلف چگونه کارآیی و هزینه را تحت تأثیر قرار می دهند.
- تمرکز کردن تنها بر روی کارآیی بدون در نظر گرفتن هزینه خیلی منطقی نیست و هنر ایجاد تعادل بین این دو است.