Vyhodnocování výkonnosti počítačových systémů ~~~~ VVPS

pozn. studijní povinnosti – na konci zkouška (písemná)
neexistuje literatura ani skripta! přednáška je kompilát.
http://d3s.mff.cuni.cz/teaching/peva
– slajdy v pdf, login performance:pe13slides
mailing list

př: ping benchmark
– server/client na jednom počítači
– RPC přes CORBu
– opakované požadavky a měření odezvy

– CORBA – na řadu věcí těžkopádná, ale funguje a používá se

dvě ~~různé~~ implementace mutexu (mutex, old futex)
– zamykání a odemykání

→ některé souvislosti se ukáží až při opravdu ~~velkém~~ množství pokusů – dočasné fáze systému
– výsledek často nelze interpretovat jen pomocí průměru
– někdy se výsledky liší mezi ~~ně~~ dvěma spuštěními (běhy) – postupná stabilizace
– benchmarku i bez zjevných změn v systému
– ~~~~ příčinou může být např. memory layout → využívání cache apod.

– cíl vzhledem k výkonnosti
- systém s maximálním výkonem
- systém s požadovanou výkonností za minimální cenu
  – otázka škálovatelnosti (stačí jen víc železa?)
- realtime systém – reakce v pevně daném čase (ne ~~nutně~~ max výkon)
- systém, kde chceme snadno analyzovat worst execution time

– během ~~~~ návrhu
  - modelování – matematický/statistický model pomocí formalismů
    – abstrakce známých charakteristik systému a prostředí
    – systémy hromadné obsluhy, petriho sítě, ...
  - simulace
    – analýza je často nemožná/náročná než řešit numericky
– během implementace a testování
  - měření (co, jak, proč) – benchmarking
  - workload relevantní pro daný systém
  - monitoring – sběr dat
  - profiling – kde ~~~~ v kódu se tráví nejvíce času?

- výhody modelování
  - odhalení problémů během design time → úspora
    - např. soupeření o prostředky (centrální databáze)
  - opakovatelnost
  - analýza extrémních / krajních příkladů - přihlášení tisíce uživatelů najednou, apod.
  - hledání vzájemných souvislostí různých elementů - zjištění, na co se zaměřit
  - relativně nejrychlejší a nejlevnější
- nevýhody modelování
  - snadno se vytvoří příliš komplexní model, který nelze analyzovat s dostupným výpočetním výkonem - je nutné hodně abstrahovat
  - příliš mnoho abstrakce může vést k zanedbání důležitých faktorů
  → nižší důvěryhodnost výsledků modelování (hlavně numerické)
    - spíš se snaží o zachycení vztahů mezi faktory
- simulace
  - doplnění dosud neimplementovaných částí systému
  - výhody
    - snížení nákladů
    - více detailů než modelování
    - (slajdy)
  - nevýhody
    - problém udržet délku simulace v rozumných mezích
    - zjednodušení → výsledky nemusí být reprezentativní
    - pokud je realistické, nemusí být opakovatelné
    - většinou pomalejší a dražší než modelování
    - náchylné na (nezjevné) chyby (např. vadný generátor náh. čísel)

- kombinace modelování a simulace
  → potvrzení (odhalení chyb v analýze)
  → kompletnost - modelování neodhalí vliv hw implementace
  - simulace nemusí odhalit chování v extrémních případech

Měření výkonu

- metriky - použitelné i na simulaci
- cíle
  - porovnání alternativ — hardware: podmínky, poměr cena/výkon, omezení
    software: optimalizace na nejlepší/postačující výkon, cenu...; může zanést na HW.
  - vliv nové fičury
  - tuning systému (např. pro konkrétní hw)
  - měření relativního výkonu (měli vrstevní aplikace)
  - apod. performance debugging
  - odhad chování v nejhorších případech

- porovnávání alternativ
  - nejspolehlivější měření podle reálných aplikací
  - nelze ale #aplikace × #HW kombinace
  - měření výkonnosti jednotlivých komponent
    - proveditelnější, ale méně spolehlivé
    - jednoduchý model → sjednocení výsledků
    - př: Open Source Benchmarking project - „platform benchmark" - zahrnuje operace
      - odhad výkonnosti na dosud neexistujících platformách - snaha o odhadnutí závislostí komponent

- vliv nové fičury
  - dvě verze systému odlišné jen v jediné věci („před a po")
  - nastavení kompilátoru apod. - také může mít smysl testovat

- ladění systému
  - hledání optimálního nastavení systému pro konkrétní prostředí
  - nastavení dlouhých, ... velikosti bufferů, cache, nastavení plánovače, počet vláken
  - možnosti se mohou vzájemně ovlivňovat, někdy velmi neintuitivně
  - typicky nelze testovat všechny kombinace (mnoho hodnoty parametrů,...)
  - experiment design theory - odhady vzájemných závislostí komponent,...
  - př. Shell Project

- škál
  - počáteční testy experiment design theory
  - odhad závislostí možností na základě experimentů
  - na základě toho identifikovat nastavení kritické pro výkonnost
  - validovat a aktualizovat výsledky dalšími experimenty
  - ostatní nastavení - randomizovat, zabránit továrnám dalších vlivů
  - prohledávání prostoru nastavení (lokálně) - viz optimalizační metody

- měření relativního výkonu
  - změna mezi dvěma verzemi systému

  - př. regression testing - zda nová verze dělá to samé (co se výkonu týče)
    - je dobré dělat to pravidelně, lze automatizovat (ale dost těžko)
    - → zpomalení se objeví poměrně brzy po vzniku
    - př. projekt BEEN

- ladění výkonu
  - zrychlování hotového programu
  - identifikace "pomalých" nastavení
  - menší benchmarky odhalující daný problém
  - profilování - které části jsou nejpoužívanější? → prioritizace
  - příčiny nemusí být zjevné
    - zpomalení → benchmark a hledání menších změn výkonu
    - hledání korelace s událostmi v systému
      - cache misses, page faults, GC, context switches, ...
  - př. performance explorer
    - profiling
    - trace alignment - během několika běhů měření různých událostí (každý nelze najednou)
    - trace vizualization
- odhad nejhorších případů (důležité pro realtime systémy)
  -
  - certifikace - "manuální" procházení kódu a odhad max doby provádění - nutná perfektní znalost všech podmínek, hw, ...
  - často nutno upravovat program, aby byl testovatelný
  - optimalizace pro worst case - pokud změna tento neoptimalizuje, změnu neprovádět - jinde se nesnaží až dost často na to, aby se to dal měřit
  - př. Metronome Real-time GC

- měření výkonu - výsledky jednodušší?
- typicky - počet událostí
  - doba trvání / mezi událostmi
  - podle výsledku

- systém dostane požadavek na službu a může
  - splnit požadavek korektně
  - splnit požadavek nekorektně → metrika spolehlivosti pro nějaké druhy chyb
    - pst chyby (dle velikosti, např. poškození paketu)
    - čas mezi chybami
  - odmítnout splnění
    - ↓ jak dlouho trvá stav, kdy se tohle děje?
    - čas mezi takovými událostmi (někdy se hodí zkusit systém resetovat, pokud nedojde do určité doby)

- analýza spolehlivosti
  - hw a sw, dost složené
  - regresní testy neodhalí vše, chyby jsou často zcela náhodné
- měření může odhalit chyby
  - např. „příliš rychlé" vykonání → [illegible] chyba, obsloužení výjimky namísto normálního provádění kódu

př. No no Regression Benchmarking
  - netriviální!
  - občas může systém padat, to by měl ale nemělo zastavit v testování
  - testy spolehlivosti jsou náročné ⇒ dát k dispozici, pokud opensource
    - nemůžou nahradit regresní testování, porovnání je podstatně složitější (i nároky na stabilní prostředí)

- analýza výkonu
  - čas odezvy (tzn. mezi přijetím požadavku a odpovědí)
    - měřit „hmatatelné výsledky"
  - produktivita
    - propustnost systému - kolik jednotek dat / čas se podařilo zpracovat
    - hůře se odhadují rozumné meze hodnot
  - využití zdrojů - utilization
    - měří, kolik % času je prostředek vytížený nad určitou mez
    - → odhalování úzkých hrdel systému anebo předimenzovaných komponent

VVPS

Metriky

AA

- měření výkonu – v jakých jednotkách?
- typicky – počet událostí
  – doba trvání / mezi událostmi
  – podle výsledku

- systém dostane požadavek na službu a může
  - splnit požadavek korektně
  - splnit požadavek nekorektně → metrika spolehlivosti pro nějaké druhy chyb
    - pst chyby (dle velikosti, např. poškození paketu)
    - čas mezi chybami
  - odmítnout splnění
    ↓
    - jak dlouho trvá stav, kdy se tohle děje?
    - čas mezi takovými událostmi (někdy se hodí zpomalit systém / resetovat, pokud nedobíhá do určité doby)

- analýza spolehlivosti
  - hw a sw, dost těžké
  - regresní testy neodhalí vše, chyby jsou často zcela náhodné
- měření může odhalit chyby
  - např. „příliš rychlé" vykonání → chyba, obsloužení výjimky namísto normálního provádění kódu

př. MonoRegression Benchmarking
  - netriviální!
  - občas může systém padat, to se ví, ale nemělo by zastavit v testování
  - testy spolehlivosti jsou náročné ⇒ dát k dispozici, pokud opensource
    - nemůžou nahradit regresní testování, porovnání je podstatně složitější (i nároky na stabilní prostředí)

- analýza výkonu
  - čas odezvy (tzn. mezi přijetím požadavku a odpovědí)
    - měřit „hmatatelné výsledky"
- produktivita
  - propustnost systému – kolik jednotek dat / čas se podařilo zpracovat
  - hůře se odhadují rozumné meze hodnot
- využití zdrojů – utilization
  - měřit, kolik % času je prostředek vytížený nad určitou mez
  → odhalování úzkých hrdel systému anebo předimenzovaných komponent

VVPS

- konzistence - jednotky míry by měly mít stejný význam na různých systémech a jejich konfiguracích
  - př. MIPS, MFLOPS nesplňují (RISC/CISC apod.)
    ↓
    wetstone, dhrystone benchmarky

- nezávislost - neměla by být optimalizovaná na konkrétní prostředí ani naopak prostředí na benchmark (což výrobci někdy dělají) → SPEC
  → ztráta výpovědní hodnoty

- př. metriky pro výkon procesorů a systémů
  - frekvence → nespolehlivá, nelineární (různé architektury, různé trvání jinde, ...)
  - MIPS, MFLOPS - o něco lepší než MIPS, protože je dobře definovaný
    \ vhodný jen pro systémy, kde na FLOPech záleží
  - SPEC - sada standardizovaných benchmarků → vyšší vypovídající hodnota
  - QUIPS
  - doba běhu

- SPEC
  - sada benchmarků, + jejich výsledků geom. průměr (aritm. průměrem dominují extrémy, medián je zas více nevyhovující)
    ovšem - není lineární vůči době běhu programu
    - není intuitivní
  - spolehlivost je dobrá pro jednotlivé benchmarky, ale méně vypovídající pro praxi
    - o něco lepší pro application benchmarks - př. SPECjvm2008
  - nezávislost: není ideální, řada výrobců optimalizuje pro SPEC, aniž by se tím výrazně zrychlovaly reálné aplikace
    - zlepšuje se na vyšších úrovních a př. vyšším počtu benchmarků
    - base a peak profil (u base jen základní přepínače překladače)
    - občas se benchmarky obměňují
    - vyvíjet na základě sady benchmarků není nutně podvodné, jen je nutno se vyhnout zjevně neobecným optimalizacím

- Virtualizační benchmarky
  - měření schopnosti HW + hypervisora spouštět více VM s různým vytížením
  - standardní průmyslové benchmarky nejsou lineární a nepoužívají dobré metriky
  - VMmark
    - sada benchmarků nad VM a bez virtualizace („reference")
    - postupně přidáváme VM
    - měří se propustnost
  - TNT
    - reference a VM
    - měří se
      - celková propustnost
      - průměrné zhoršení response time
  - pareto curves
    - graf odezva / throughput
    - pareto-optimální servery
      - je jich víc možných, záleží na prioritách

- Execution time
  - intuitivní, jednoduchá
  - pozor na background load
  - čas CPU?
    - nevypovídající o celkovém chování
    - blbě se měří (interrupt based accounting)
    - ignoruje nezbytné čekání
    - také ovlivněno background loadem (TLB, cache, ...)
  - kompromis: CPU & wall clock time
  - splňuje většinu požadavků
    - problém s jednoduchostí měření – přesnost
    - opakovatelnost – pouze statisticky
    - → viz pozdější přednášky

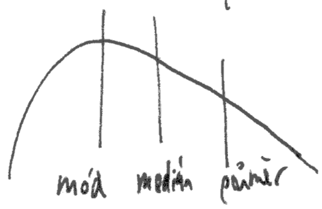
- QUIPS - Quality improvement per second
  - matematický problém řešený iterativní metodou
  - Kvalita = jak blízko je současný stav k výsledku (v každém kroku se zlepšuje)
  - HINT benchmark: $\int_0^1 \frac{(1-x)}{(1+x)} dx$
    - postupné zpřesňování integrálu
    - kvalita = $\frac{1}{\text{horní mez} - \text{dolní mez}}$
    - vypovídající hodnota je omezená - specifický numerický problém
    - jak vytvořit analogický benchmark pro I/O, apod.?
  - nenechá se snadno rozházet náhodnými vlivy
  - měří se výkon, ne činnost - doplňuje FLOPS
- praxe - při velkých zákazích
  - vytvořit si benchmark na míru
  - zvečinit výrobcům, nechat je optimalizovat systém na benchmarky
  - smluvně vyžadovat opakovatelnost

metriky — úsilí na dosažení cíle (MIPS, FLOPS, ...) - inherentně nespolehlivé
cíl (QUIPS apod.)

- nedeterminismus v době běhu
  - warmup - několik prvních běhů typicky trvá (výrazně) déle
    ⇒ vynechat ze statistik
- předpoklady (které ale někdy neplatí) (skryté závislosti) ⇒ náhoda je nepochopení souvislostí
  - doby běhu jsou náhodné
  - nezávislé
  - rovnoměrně distribuované

→ běžné statistické metody

- často požadavek na jedno číslo
  - u jednoduchých benchmarků se takto někdy shrnout dají
- výběr statistické metody
  - distribuce nemusí být rovnoměrná, může být multimodální (víc lok. maxim), ...
  - měla by mít intuitivní výklad, něco opravdu znamenat
  → medián, průměr, ...
- doby běhů → náhodné proměnné $X_1, \ldots$ - neznáme jejich distribuce
  - $x_1, x_2, \ldots$ - realizace náh. prom. (naměřené hodnoty)
- prerekvizity: distribuční fce, fce hustoty psti, disk. a spoj. NV, kvantily, stř. hodnoty



→ co vybrat?
- kategorická data → mód
- součet dat dává (fyz.) smysl? → aritmetický průměr
- vychýlená distribuce → průměr nereprezentativní, → medián
- jinak → průměr

- problémy metrik
  - průměr - <u>velmi</u> citlivý na extrémní výchylky
    → ořezaný průměr (kromě několika % krajních výsledků) - opatrně
  - medián - naopak ignoruje hodně informací - extrémní informace
    - vícemodální data se zhruba stejně velkými clustery ⇒ fragilní
  - pozor na korelace → nelze násobit průměry
  - geometrický průměr: $\sqrt[n]{\prod_{i=1}^{n} x_i}$
    - použít pokud má smysl součin ~~násobek~~ hodnot (místo součtu)
    - vícedimenzionální stoly → kostka
    - průměrné zrychlení ve víceosém systému
    - nezávislý na base → poměry (stále je potřeba, aby geom. průměry čitatele a jmenovatele měly význam!)
- jiné průměry - nutno pečlivě vybírat
  - agregační metriky
- linearita, spolehlivost!

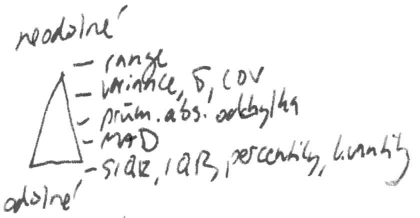
- harmonický průměr
  - „fyzikální" význam: doba obsluhy požadavku – pozor na obecný trend u serverů
  - ...
- poměry
  - podíl různých hodnot dává smysl ⇒ podíl sum
  - konstanta v čitateli ⇝ harmonický průměr
  - „skoro konstantní" poměry → geom. průměr
    - na průměrných benchmarcích, apod.
    - vhodné při snaze o odhadnutí nějaké konstanty: $a_i = c \cdot b_i$
  - př: průměrné zrychlení velikosti zdroj. kódu (před optimalizací a po ní)
    - program / kB před / kB po / poměr $\frac{\text{po}}{\text{před}} \cdot 100$
    - součet čitatelů i součet jmenovatelů dává smysl, nabízí se spočítat velikosti a spočítat jejich podíl = podíl průměrů
      → průměrují se programy různých velikostí, každý kB má různý vliv (dominují velké programy)
    - dává výběr programů smysl? Optimalizace mají korelovat s velikostí programu, což nezohledňuje
    - ideální by bylo provážit velikosti programů s frekvencí jejich výskytu – neznáme
    - radši: průměrovat přes programy
    - geom. průměr – výsledky v blízkém okolí
      → uhodnější
      – stále nezohledňuje velikost programu v zrychlení
  - př: average speedup:
    - benchmark / ref. time / measured time / speedup
    - součty a průměry časů opět dávají smysl
    - benchmarky mají ale velmi odlišné hodnoty
    - geom. průměr nelze použít – velký rozsah hodnot
      model $a_i = c * b_i$ nesedí – velká škála různých faktorů
    - spíš jej používají, ale v principu to není dobře
    → ideálně přiřadit váhy benchmarkům
    - jinak, můžeme-li předpokládat, že naměřené hodnoty jsou reprezentativní, použít harmonický průměr) zrychlení
    - není dobré průměrovat podíl. odlišná čísla

- vážený průměr
  - součet vah vždy = 1
- na tohle můžou být ve zkoušce úlohy

- variabilita u nedeterministických měření
  - také (spolu s průměrem) vypovídá o kvalitě systému
  - histogram – potřeba vhodně zvolit jemnost (R dělá heuristiku)
  - pozor na warm-up – vyříznout

- se statistikou nelze pracovat moc dobře algoritmicky
- každé rozhodnutí velmi přesně zdůvodnit

- metriky variability dat
  - průměrná absolutní odchylka
  - mediánová absolutní odchylka (MAD)
  - mezikvantilový interval (IQR)
  - semi-interquantile range
  - 10.-90. percentil (ořezání extrémů)
  - variance, směrodatná odchylka, COV
- výběr metriky závisí na
  - požadované robustnosti vůči okrajovým hodnotám:



  - použité metriky benchmarku
  - konkrétních datech

↓ víc není vždy lepší; outliers mohou mít velký význam

- rozsah
  - pokud proměnná má smysluplné hranice
  - pokud měříme hard-realtime systémy
- rozptyl
  - outliers vzácné a ne příliš velké
  - pokud je distribuce blízko unimodální symetrické
  - pokud je jako metrika centr. tendence použitý průměr
  - upřednostňovat COV
- průměrná absolutní hodnota
  - std. odchylka není příliš robustní vůči okrajovým hodnotám

wps

- robustní metriky
  - pokud se moc mění mediány,
  - pokud je distribuce multimodální, mají hodně okrajových hodnot apod.
  - kvantily a IQR jsou různé, kvantily mají smysl pro soft-realtime systémy

- obecně při vyhodnocení záleží na aplikaci a cíli měření
  - také na dalším vyhodnocování, s některými metrikami se špatně pracuje
  - např. chceme odhadovat distribuci

- Jak interpretovat a reportovat variabilitu ve výsledcích?
  - systém má vlastnosti, které neznáme, ač mohou být deterministické
  - měřící nástroje, prostředí, ve kterém se měří → chyby, šum
  - termíny ← accuracy – jak se měření blíží reálné hodnotě
    - v češtině se termíny zaměňují, pozor!
    - accuracy „správnost" – jak se měření blíží reálné hodnotě
    - precision „přesnost" – množství šumu v naměřených hodnot (jejich stabilita)
    - resolution „rozlišení" – nejmenší krok, který jsme schopni změřit
- chyby v měření
  - systematické
    - ovlivňují správnost
    - měly by být eliminovány
      - např. overhead timeru apod.
      - velmi opatrně vše nastavit, zkontrolovat
      - minimalizace rušení (~~…~~ služby, uživatelé, …)
      - stabilní prostředí (switchu updaty)
      - randomizace parametrů prostředí
        → převedení na náhodné chyby
  - náhodné
    - ovlivňují přesnost
    - nedeterministické, nepředvídatelné – často nelze kontrolovat
    - mají stejnou pst. zhoršení / zlepšení měření!
    - reportování vše dohromady, většinou nelze izolovat
- model náh. chyb
  - předpoklady v praxi mnohdy neplatí (nebo zdaleka ne všechny), přesto jsou modely užitečné
  - normální rozdělení
    - střední hodnota – μ: přidává k ní systematická chyba
    - náhodné chyby ovlivňují rozptyl σ^2
- vždy reportovat alespoň naměřenou hodnotu ± odchylka
  - nezveřejňovat nic zjevně špatného nebo nepoužitelného
  - též význ. číslice ~~…~~ chyby maximální
- „sigma rules"
  - ~ 68% hodnot v hranicích chyby měření od stř. h.
  - ~ 95% hodnot $\in (\mu - 2\sigma, \mu + 2\sigma)$
  - ~ 99,7% hodnot $\in (\mu - 3\sigma, \mu + 3\sigma)$

- jak jsou naše odhady spolehlivé?
- intervaly spolehlivosti: opravdová střední hodnota je ve nějakém intervalu se zvolenou pstí
- nepřímá úměra mezi přesností odhadu a spolehlivostí
- o naměřených x_i průměrech je nutno uvažovat jako o NV

x_i i.i.d:

$E(X_i) = \mu \rightarrow E(\bar{X}_i) = \mu$

$var(X_i) = \sigma^2 \rightarrow var(\bar{X}_i) = \frac{\sigma^2}{n}$

$sd(X_i) = \sigma \rightarrow sd(\bar{X}_i) = \frac{\sigma}{\sqrt{n}}$

(sd = standard deviation)

standardní odchylka aritmetického průměru
- „standardní chyba"
- lze využít do výsledku (spolu s n)

- interval spolehlivosti je počítaný vždy pro konkrétní měření
  - $pro(\alpha)$ ... pravděpodobnost, že průměr neleží v intervalu, je α
  - pro jiné měření jiná spolehlivost!

... viz slidy ...

wps

Porovnávání alternativ

- dva benchmarky, jejichž výsledky se „překrývají"
- nepárované
  - dvě sady dat z různých systémů, ev. různé velikosti
  - chceme porovnat neznámé průměry
  - předp: variabilita = náh. chyby
  - různé metody: confidence interval pro rozdíl, dva CI, ...
- pokud se CI obou měření nepřekrývají, lze s „rozumnou" mírou spolehlivosti předpokládat ostře větší rozdíl mezi průměry systémů
  - překrývající CI, průměr/stř.h. nejsou v CI druhého měření → statistický test, pokud nelze, nevyvozovat cokoliv
  - překrývající CI, stř. h. v CI druhého měření – nelze dokázat, zda nejsou stejné
- překryvy intervalů spolehlivosti
  - snadná metoda – implementace, vizualizace, intuitivita
  - nemusíme spoléhat na distribuci
  - není zjevná spolehlivost celkového výsledku (netriviální, někdy nelze)
  - není zjevná spolehlivost (často nedetekuje podstatné rozdíly), std chybový interval 68% příliš riskantní 95% příliš konzervativní
- kvalita porovnání
  - porovnávané hodnoty nejasné – jsou nebo nejsou stejné; nelze mluvit o poměru
  - rozhodnutí na naměřených statistikách
    - rozdíl > threshold ⇒ ...

- nulová hypotéza (H0) – „default", proti kterému hledáme důkazy
  - H0: průměry měření jsou stejné, je rozdíl, ale nejsou
  - dostatečně silné důkazy, že rozdíl není dílem náhody ⇒ odmítnout H0 ⇒ průměry jsou různé
  - dva typy chyb
    - typ 1: false positive (zamítnutí pravdivé H0)
    - typ 2: false negative (přijetí nepravdivé H0)
  - míra významnosti
    - určuje chybu 1. typu, tradeoff s chybou 2. typu
  - síla testu je často neznámá
    - záleží na míře významnosti, velikosti vzorku a reálném rozdílu průměrů

α ... míra významnosti
β ... síla testu

	H0 = true	H0 = false
zamítnout H0	type 1, $p = \alpha$	true positive, $p = 1 - \beta$
H0	true negative, $p = 1 - \alpha$	type 2, $p = \beta$

- jak zvolit míru významnosti testu?
  - záleží, co by nás stála chyba 1. typu a co získáme, když neuděláme chybu 2. typu
  - předpoklady
    - normální distribuce náh. (i.i.d.) chyb
- ~~[illegible]~~
    - stř. odchylky se mohou v měřeních lišit
    - jakékoliv data překryvu CI
  - pro zjištění 5% míry významnosti s použitím CI, je potřeba vybrat míru spolehlivosti pod 95%. Pro měření s podobnými rozptyly stačí používat hodnoty okolo 84%.
- reportování výsledků
  - odmítnutí H0 s 5% významností ... [illegible]

- nelze odmítnout H0 na hladině 5% významnosti => nenašli jsme důkazy
  - získání více dat -> lze zúžit CI

- porovnávání proporcí
  - např. % času stráveného v kódu
  - poměr - měření samplováním
- chceme CI pro poměr
- CI pro průměr binom. rozdělení lze aproximovat normálním
- dvouvzorkový test
  ~~[illegible]~~

- statistické testy
  - testová statistika počítaná z dat (např. rozdíl průměrů)
  - kritická oblast
    - pokud se výsledek testu nachází zde, zamítnout H0
    - [illegible] na CI
  - p-hodnota: nejmenší hladina významnosti umožňující zamítnout H0
    - čím menší, tím více důkazů máme pro zamítnutí H0
    - zároveň odpovídá šanci, že dostaneme tato/extrémnější data, pokud H0 platí

- srovnání
  - nepárová: průměry, stř. hodnoty, odchylky, distribuce, ...
  - párová: dvě měření pro každou položku (na stejných vstupech)
    - o něco silnější
  - více než dvě měření (populace)
    - hierarchické modely
    - ANOVA

- párové porovnání
  - převod na nepárové (rozdíl měření mít konstantní)
  - CI: kontrola, zda interval obsahuje 0
  - testy: jednovzorkové, zda prům. rozdíl je 0

- nepárové porovnání
  - CI (pro ≠ metody, pro které můžeme zkonstruovat CI)
  - statistické testy
    - T-test (Welchův t-test)
    - U-test - nečiní předpoklady o distribuci, neparametrický
      -> p-hodnota
    - Kolmogorov-Smirnov - neparametrický
      - porovnání distribuce: dvou vzorků / vzorku a reference
      -> lze užít jako test normality (ale ≠ silný)

vvps

- ANOVA
  - snaha o rozlišení různých vlivů na rozptyl výsledků měření
  - odhad jejich stat. významnosti
  - podrobnosti viz slidy

- v praxi nelze některé požadavky a předpoklady splnit → myslet!
  - i tak mohou statistické metody poskytnout cenné info
  - stat. metody nemohou nahradit analýzu systému
  - nevěřit statistice, kterou jsem si sám nezfalšoval

- velikost efektu
  - ~~[illegible]~~
  - Cohenovo D a další odhady
  - vhodné zvolit míru, kterou budeme ještě zanedbávat

- Exploratory data analysis

- „Anscombova čtyřka" úplně stejné charakteristiky (průměr, medián, rozptyl, ...).
  - viz slajdy, úplně stejné charakteristiky (průměr, medián, rozptyl, ...), ale ve skutečnosti se hodně liší → dost věcí lze vyčíst z grafu
  → vždycky se dívat na data!
- analýza dat před použitím stat. metod
  - pro odhalení struktury, outliers
  - ...
  - závislost na grafickém zobrazení
- př. SciMark2 (viz slidy)
  - FFT v C# Mono
- run-sequence plot
  - index × value
  - posuny v hodnotách (variaci, šum, outliers, multimodalita
  - interference s jiným procesem (pravidelnost outlierů)
- histogram – odhad hustoty, představa o distribuci, středu, rozptylu
  - spolehlivější než run-sequence plot, ale nezohledňuje časové závislosti
  - vyzkoušet různé jemnosti (použít matematický model na odhad vhodné jemnosti)
  - neměl by se měnit po další měření nebo dobré změny jemnosti

- nezávislost měření: lag plot
  - osy: x: naměřené hodnoty
    y: naměřené hodnoty bez prvních L } nebo naopak, např. MSSA
  - náhodnost dat, sekvenční korelace, outliers, odhad chybného modelu při časové závislosti
  - jedna hodnota => na základě jedné hodnoty nelze říct, jaká bude následující
  - data v clusteru => přeskakování mezi clustery?
  - podobné, nejde o časovou závislost (clustery na diagonále)
  - zajímavý lag -> lze vyšetřovat periodu

- bootstrap plot
  - náhodné převzorkování (lišadlový vybl. lze vyšat víceraft)
  - pro každý subsample znovu vypočítat statistiku
  - → stabilita výsledku? tvar celkové distribuce?

- Quantile-quantile plot
  - pro porovnání distribucí dvou množin dat
  - $k=4$
  - stejné ⇒ na úhlopříčce
  - porovnání s normální distribucí

- boxplot

- violinplot

- bihistogram

- Prezentace dat
  - Jain
  - - reflexivní, nemanipulovat
    - jednoduše, pochopitelně, popisně