Benchmarkování

Základy testování výkonnosti a spolehlivosti HW

Richard Lipka

Benchmarkování

Lež

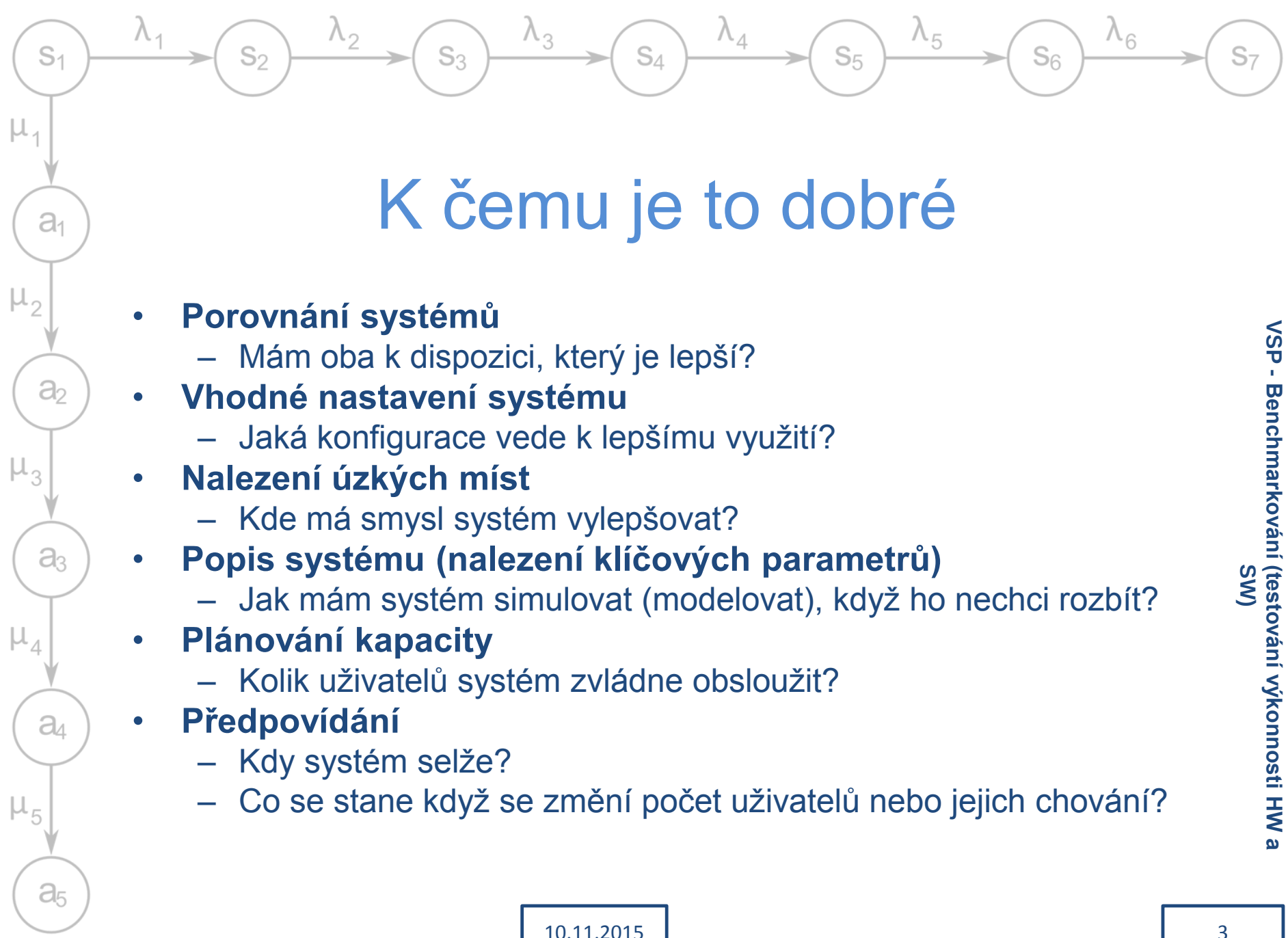
Větší lež

Statistika

Benchmarky

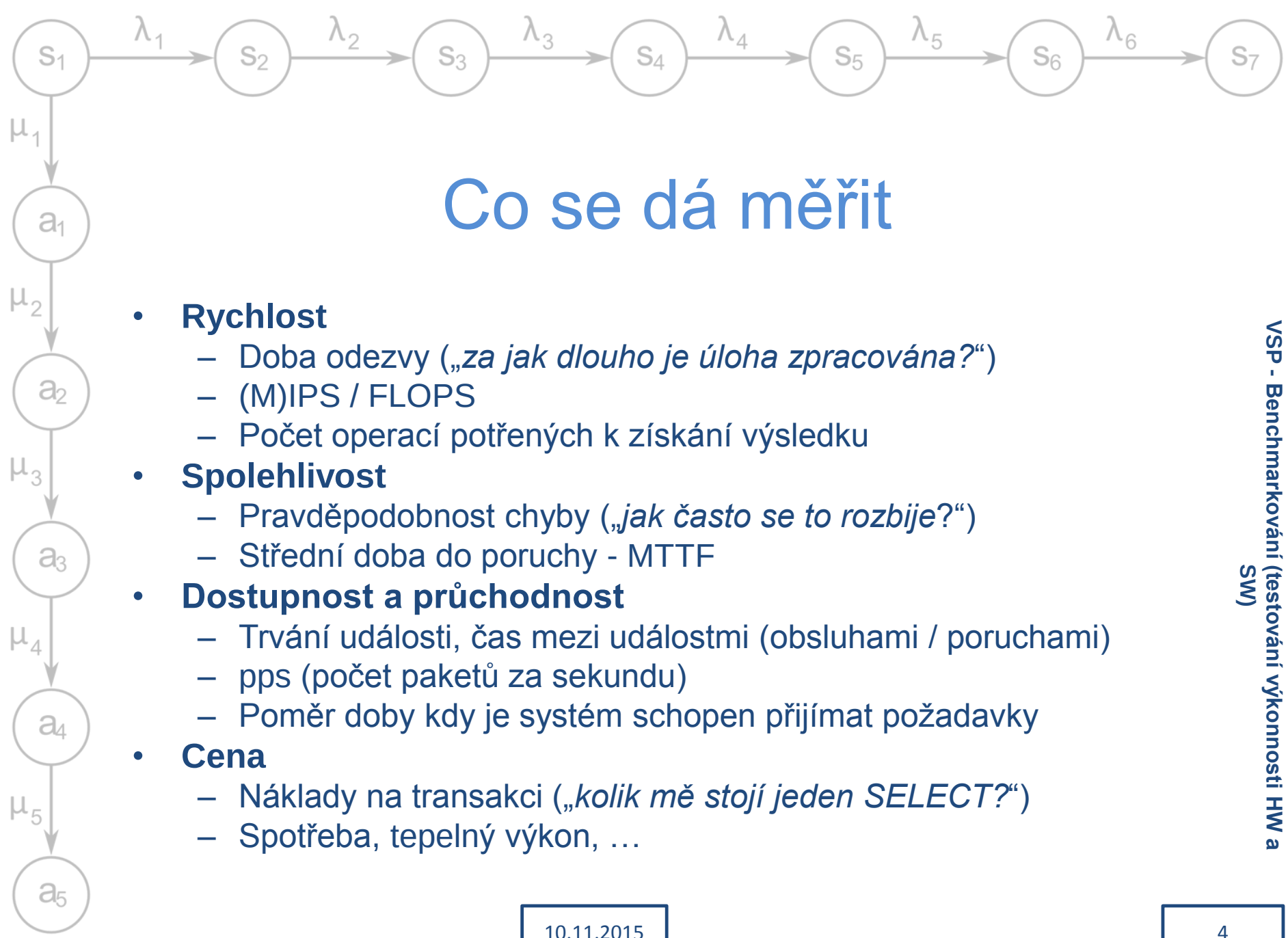
Dr. Aaron Harwood

- Standardní test (program) sloužící k hodnocení výkonu počítačového systému



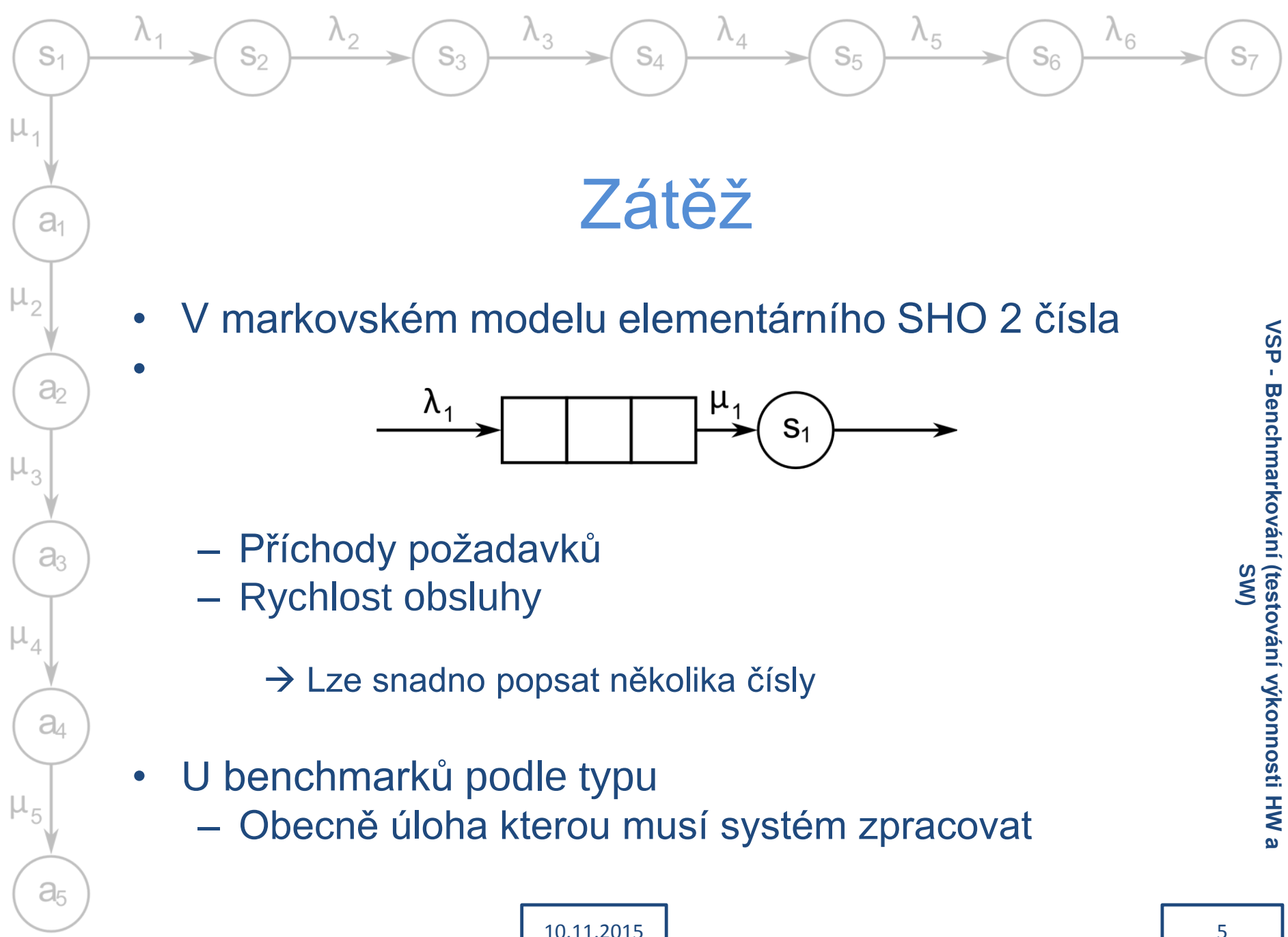
K čemu je to dobré

- **Porovnání systémů**
  - Mám oba k dispozici, který je lepší?
- **Vhodné nastavení systému**
  - Jaká konfigurace vede k lepšímu využití?
- **Nalezení úzkých míst**
  - Kde má smysl systém vylepšovat?
- **Popis systému (nalezení klíčových parametrů)**
  - Jak mám systém simulovat (modelovat), když ho nechci rozbít?
- **Plánování kapacity**
  - Kolik uživatelů systém zvládne obsloužit?
- **Předpovídání**
  - Kdy systém selže?
  - Co se stane když se změní počet uživatelů nebo jejich chování?



Co se dá měřit

- **Rychlost**
  - Doba odezvy („*za jak dlouho je úloha zpracována?*“)
  - (M)IPS / FLOPS
  - Počet operací potřených k získání výsledku
- **Spolehlivost**
  - Pravděpodobnost chyby („*jak často se to rozbije*?“)
  - Střední doba do poruchy - MTTF
- **Dostupnost a průchodnost**
  - Trvání události, čas mezi událostmi (obsluhami / poruchami)
  - pps (počet paketů za sekundu)
  - Poměr doby kdy je systém schopen přijímat požadavky
- **Cena**
  - Náklady na transakci („*kolik mě stojí jeden SELECT?*“)
  - Spotřeba, tepelný výkon, …



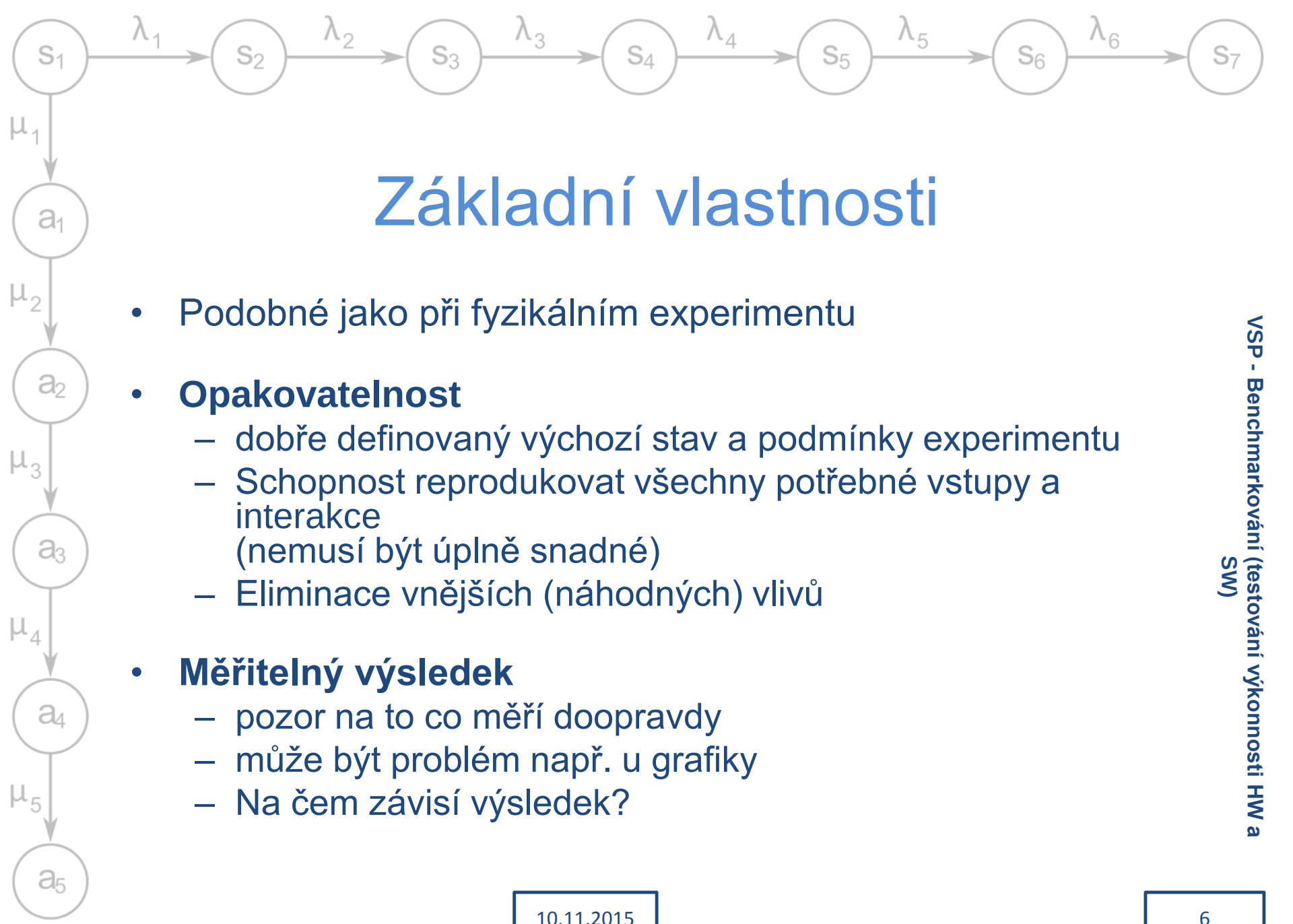
Zátěž

- V markovském modelu elementárního SHO 2 čísla
-

  - Příchody požadavků
  - Rychlost obsluhy

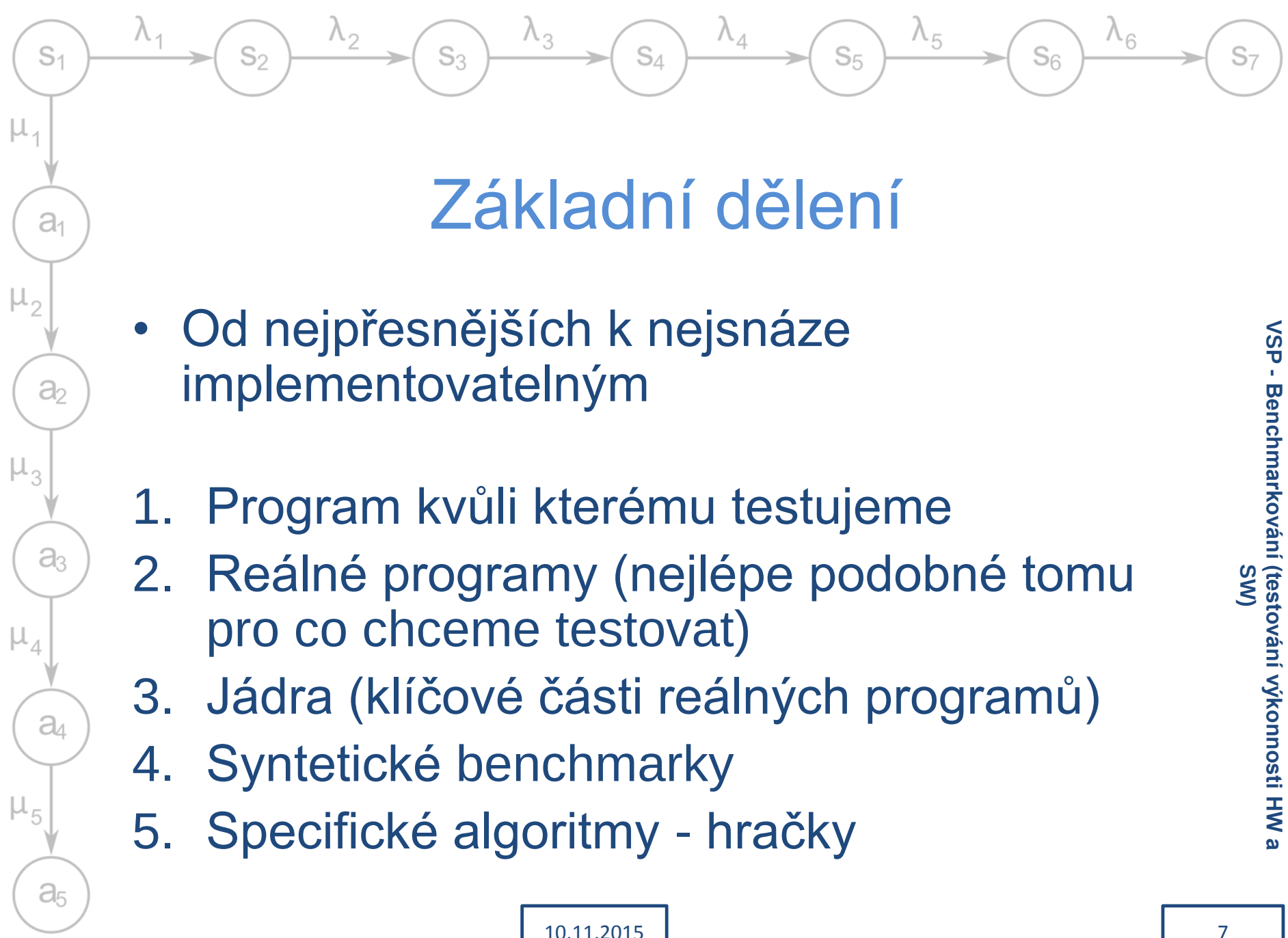
    → Lze snadno popsat několika čísly

- U benchmarků podle typu
  - Obecně úloha kterou musí systém zpracovat



Základní vlastnosti

- Podobné jako při fyzikálním experimentu
- **Opakovatelnost**
  - dobře definovaný výchozí stav a podmínky experimentu
  - Schopnost reprodukovat všechny potřebné vstupy a interakce
    (nemusí být úplně snadné)
  - Eliminace vnějších (náhodných) vlivů
- **Měřitelný výsledek**
  - pozor na to co měří doopravdy
  - může být problém např. u grafiky
  - Na čem závisí výsledek?



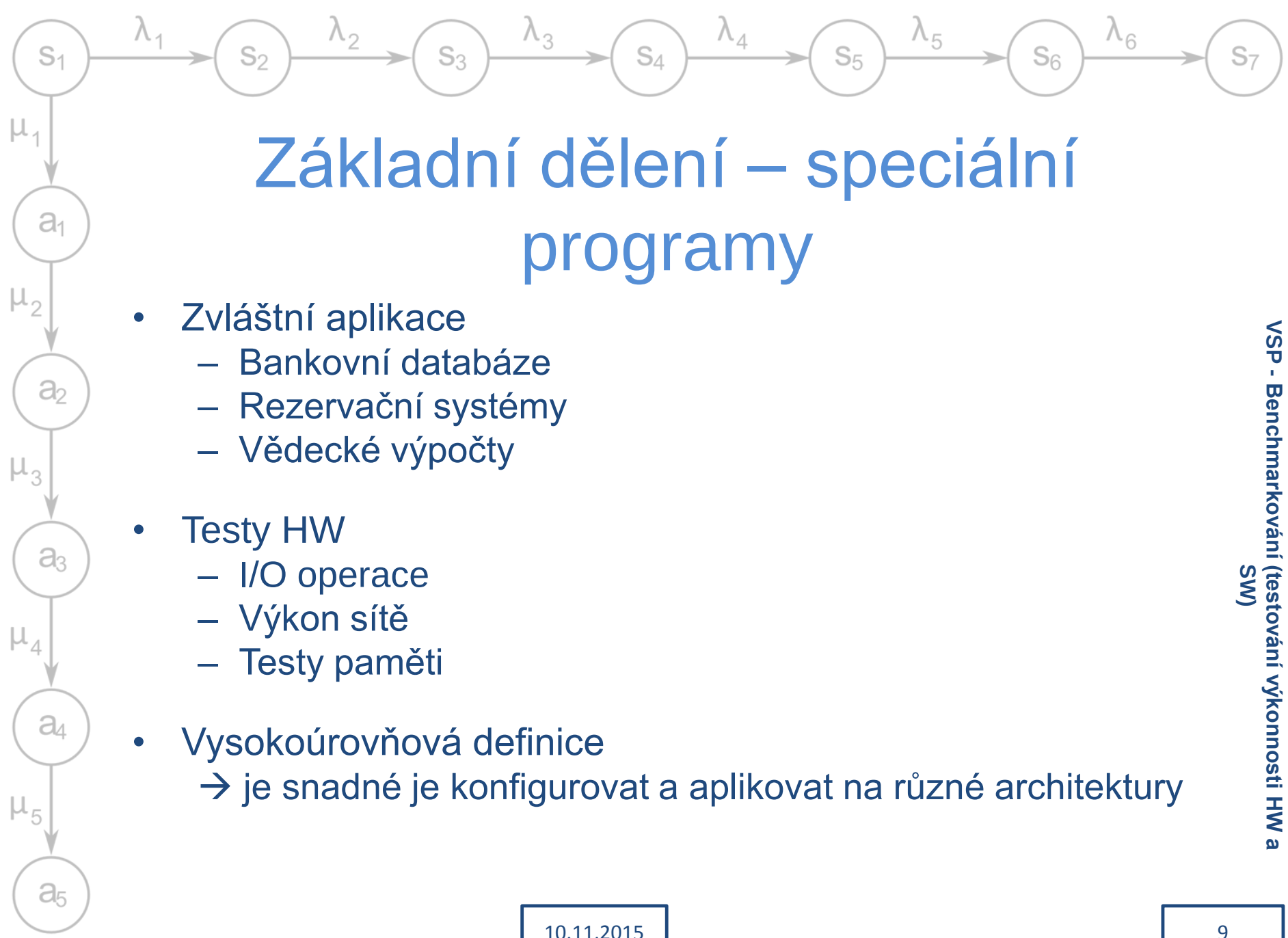
Základní dělení

- Od nejpřesnějších k nejsnáze implementovatelným

1. Program kvůli kterému testujeme
2. Reálné programy (nejlépe podobné tomu pro co chceme testovat)
3. Jádra (klíčové části reálných programů)
4. Syntetické benchmarky
5. Specifické algoritmy - hračky

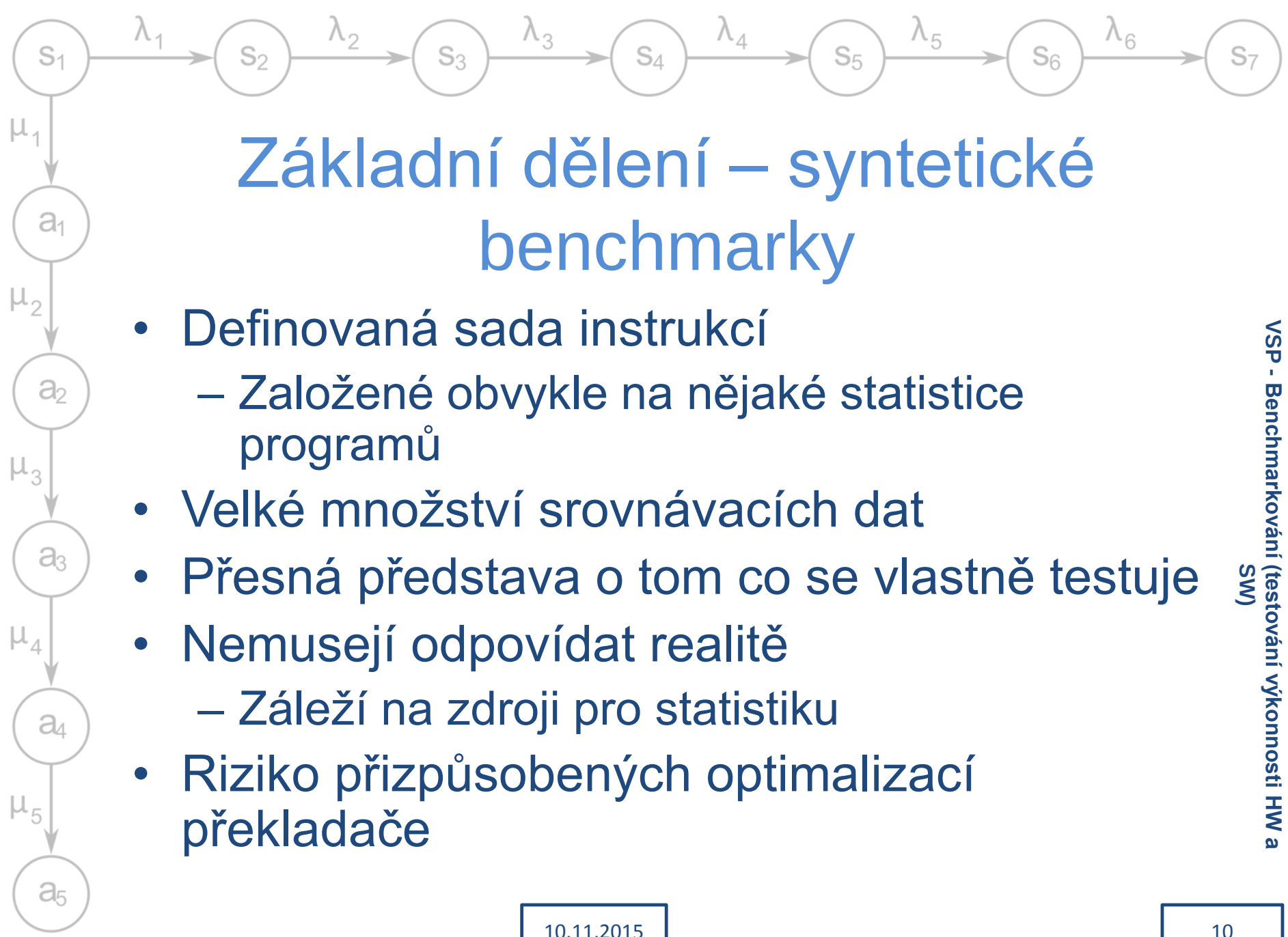
Základní dělení – reálné programy

- Lze testovat s tím co má být doopravdy použito
- Složitější příprava testů a volba zátěže, volba metriky
- Problém srovnávání více testovaných systémů



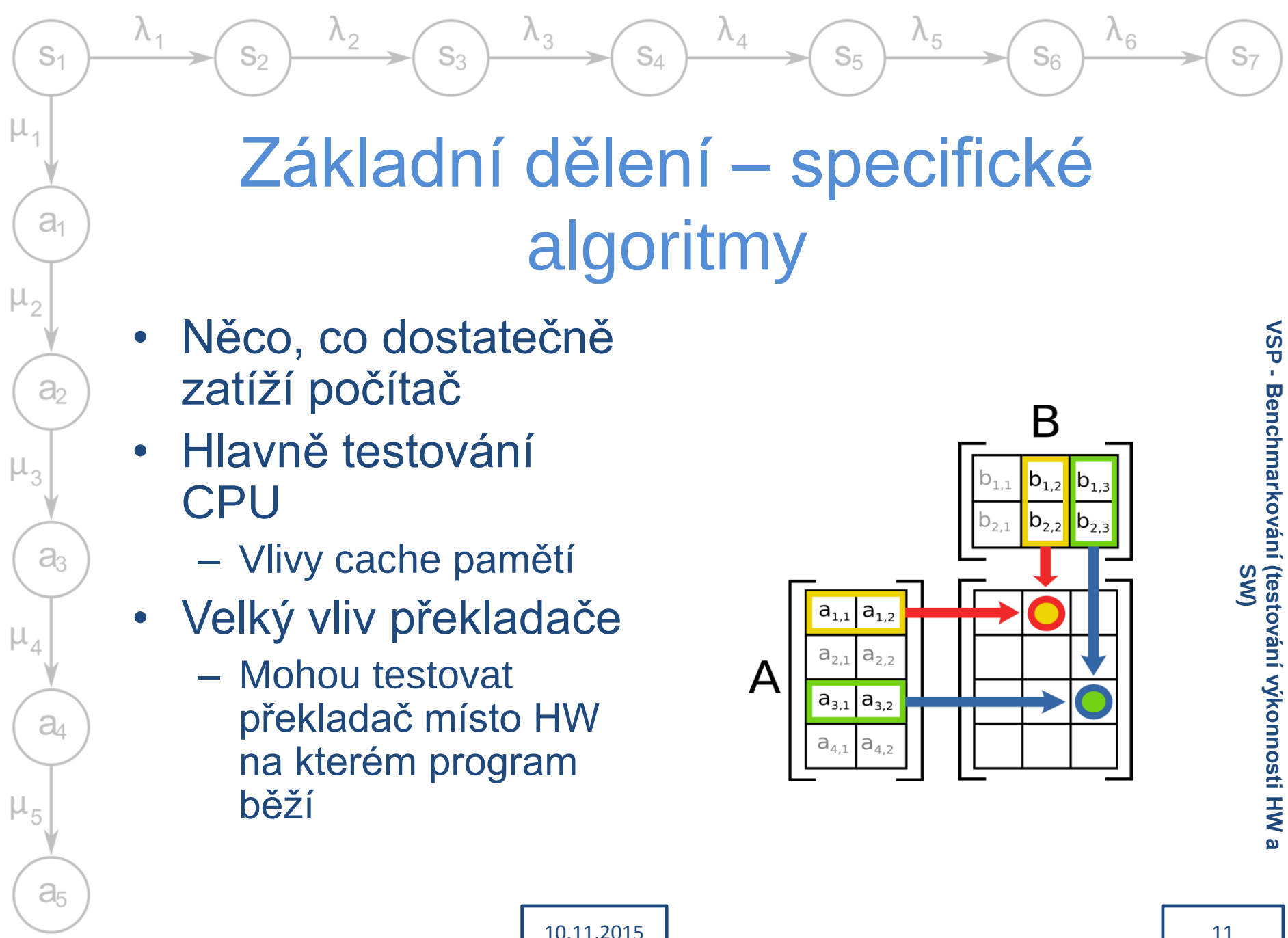
Základní dělení – speciální programy

- Zvláštní aplikace
  - Bankovní databáze
  - Rezervační systémy
  - Vědecké výpočty
- Testy HW
  - I/O operace
  - Výkon sítě
  - Testy paměti
- Vysokoúrovňová definice
  → je snadné je konfigurovat a aplikovat na různé architektury



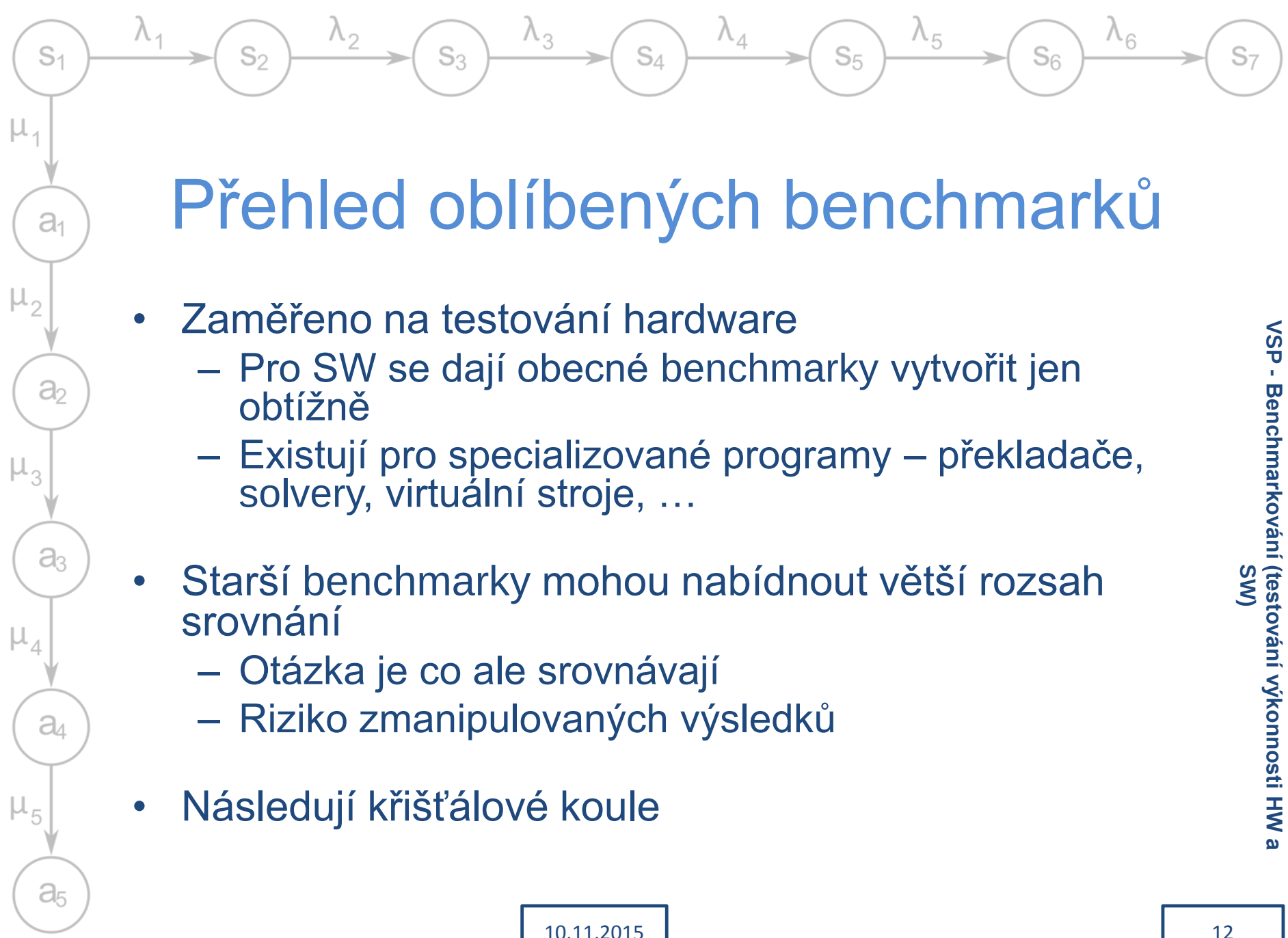
Základní dělení – syntetické benchmarky

- Definovaná sada instrukcí
  - Založené obvykle na nějaké statistice programů
- Velké množství srovnávacích dat
- Přesná představa o tom co se vlastně testuje
- Nemusejí odpovídat realitě
  - Záleží na zdroji pro statistiku
- Riziko přizpůsobených optimalizací překladače



Základní dělení – specifické algoritmy

- Něco, co dostatečně zatíží počítač
- Hlavně testování CPU
  - Vlivy cache pamětí
- Velký vliv překladače
  - Mohou testovat překladač místo HW na kterém program běží

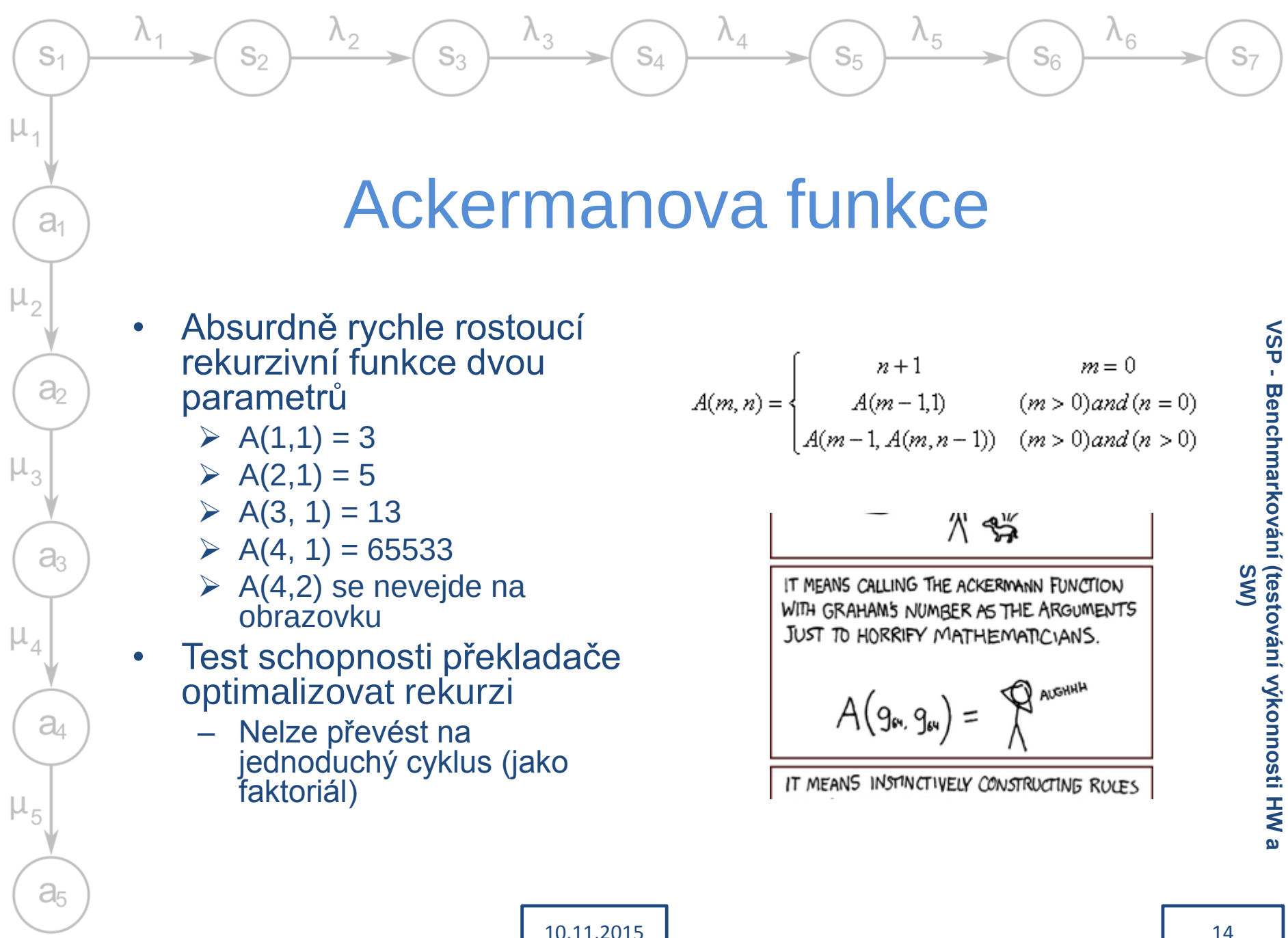


Přehled oblíbených benchmarků

- Zaměřeno na testování hardware
  - Pro SW se dají obecné benchmarky vytvořit jen obtížně
  - Existují pro specializované programy – překladače, solvery, virtuální stroje, …
- Starší benchmarky mohou nabídnout větší rozsah srovnání
  - Otázka je co ale srovnávají
  - Riziko zmanipulovaných výsledků
- Následují křišťálové koule

Erastothenovo síto

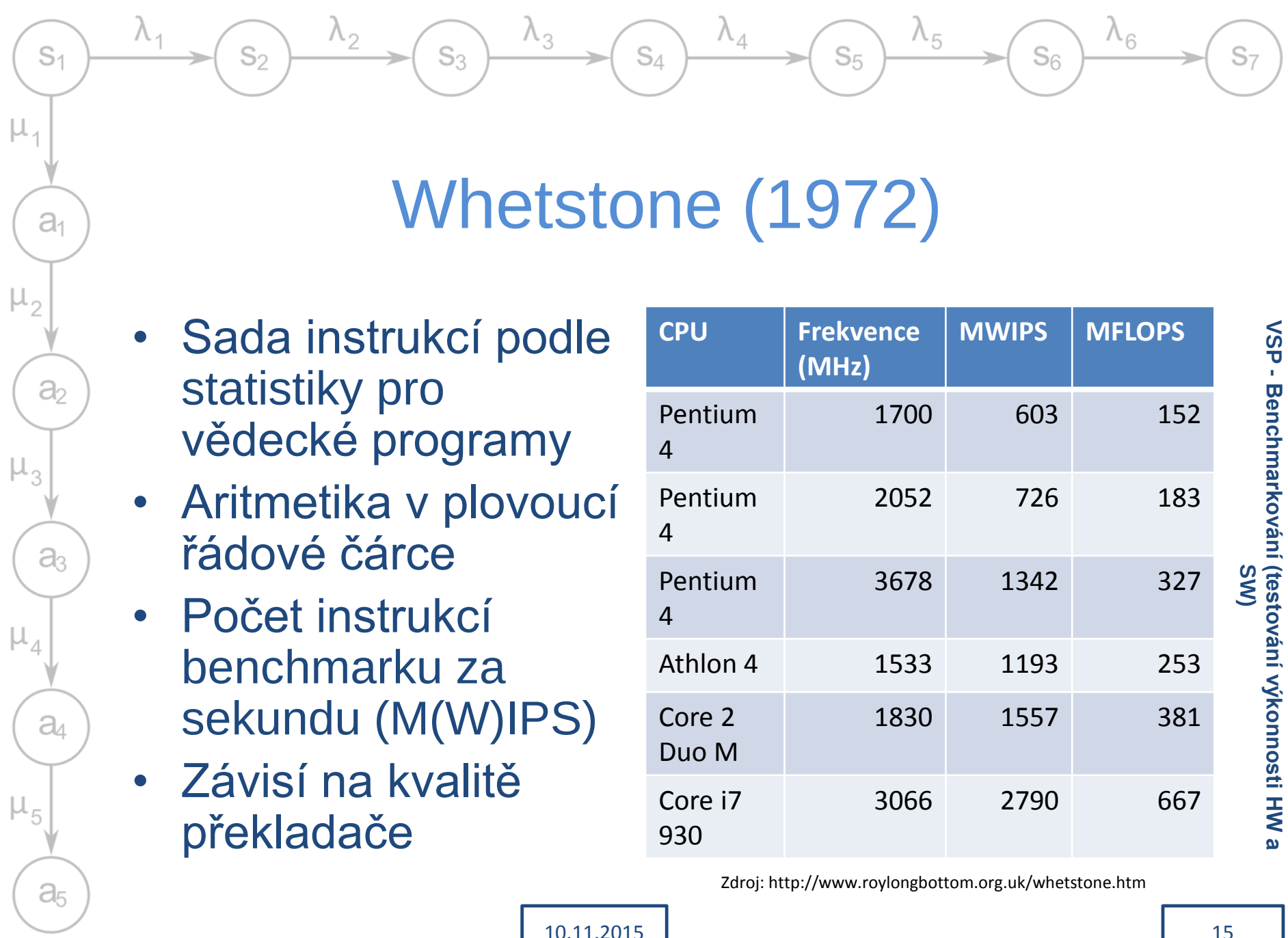
- Hledání prvočísel
- Výkon závisí na
  - Rychlosti paměti
  - Implementaci polí
  - Velikosti stránek – co je v cache a co se čte



Ackermanova funkce

- Absurdně rychle rostoucí rekurzivní funkce dvou parametrů
  - A(1,1) = 3
  - A(2,1) = 5
  - A(3, 1) = 13
  - A(4, 1) = 65533
  - A(4,2) se nevejde na obrazovku
- Test schopnosti překladače optimalizovat rekurzi
  - Nelze převést na jednoduchý cyklus (jako faktoriál)

$$A(m,n) = \begin{cases} n+1 & m = 0 \\ A(m-1,1) & (m > 0) and (n = 0) \\ A(m-1, A(m,n-1)) & (m > 0) and (n > 0) \end{cases}$$

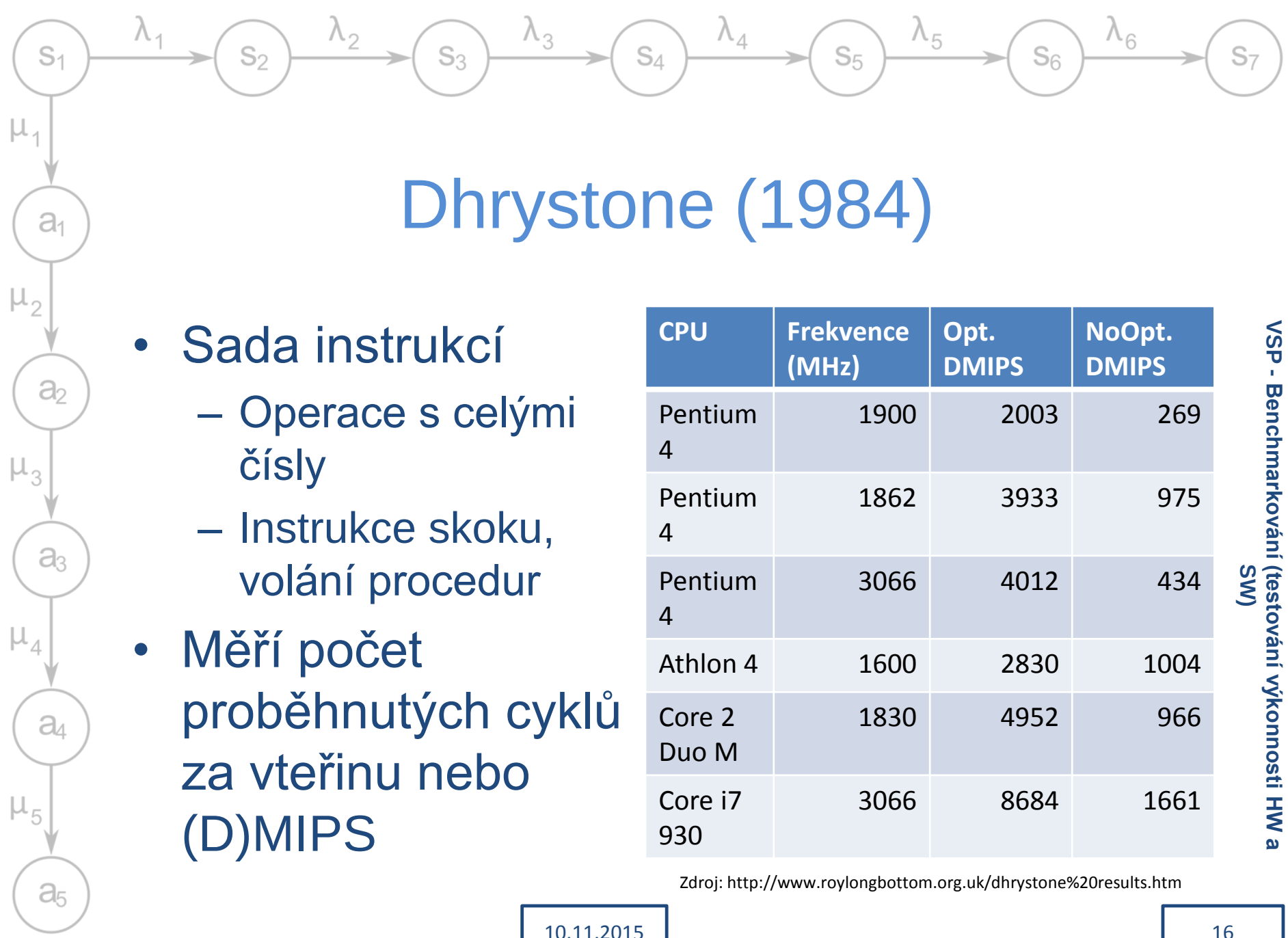


Whetstone (1972)

- Sada instrukcí podle statistiky pro vědecké programy
- Aritmetika v plovoucí řádové čárce
- Počet instrukcí benchmarku za sekundu (M(W)IPS)
- Závisí na kvalitě překladače

CPU	Frekvence (MHz)	MWIPS	MFLOPS
Pentium 4	1700	603	152
Pentium 4	2052	726	183
Pentium 4	3678	1342	327
Athlon 4	1533	1193	253
Core 2 Duo M	1830	1557	381
Core i7 930	3066	2790	667

Zdroj: http://www.roylongbottom.org.uk/whetstone.htm

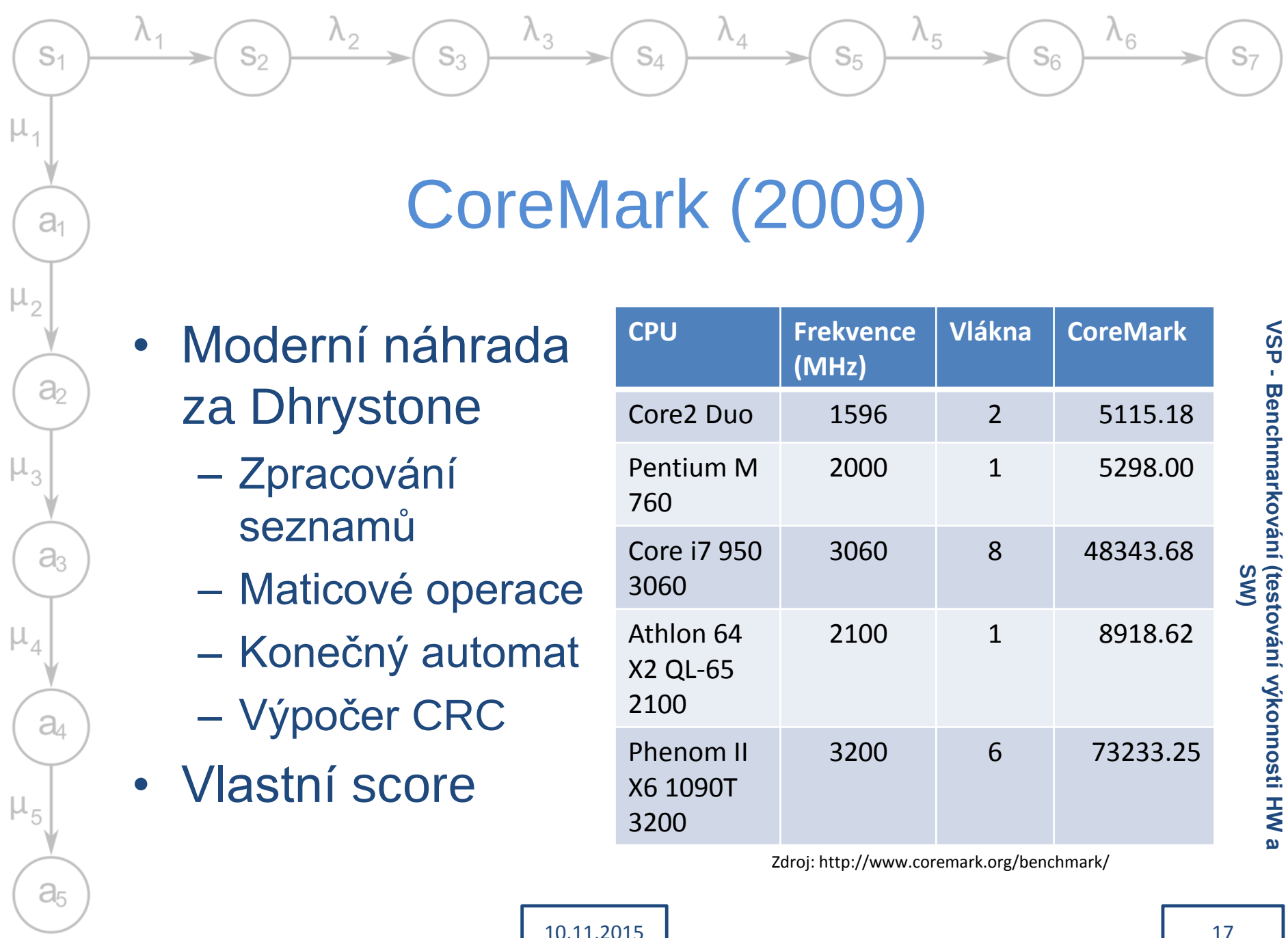


Dhrystone (1984)

- Sada instrukcí
  - Operace s celými čísly
  - Instrukce skoku, volání procedur
- Měří počet proběhnutých cyklů za vteřinu nebo (D)MIPS

CPU	Frekvence (MHz)	Opt. DMIPS	NoOpt. DMIPS
Pentium 4	1900	2003	269
Pentium 4	1862	3933	975
Pentium 4	3066	4012	434
Athlon 4	1600	2830	1004
Core 2 Duo M	1830	4952	966
Core i7 930	3066	8684	1661

Zdroj: http://www.roylongbottom.org.uk/dhrystone%20results.htm

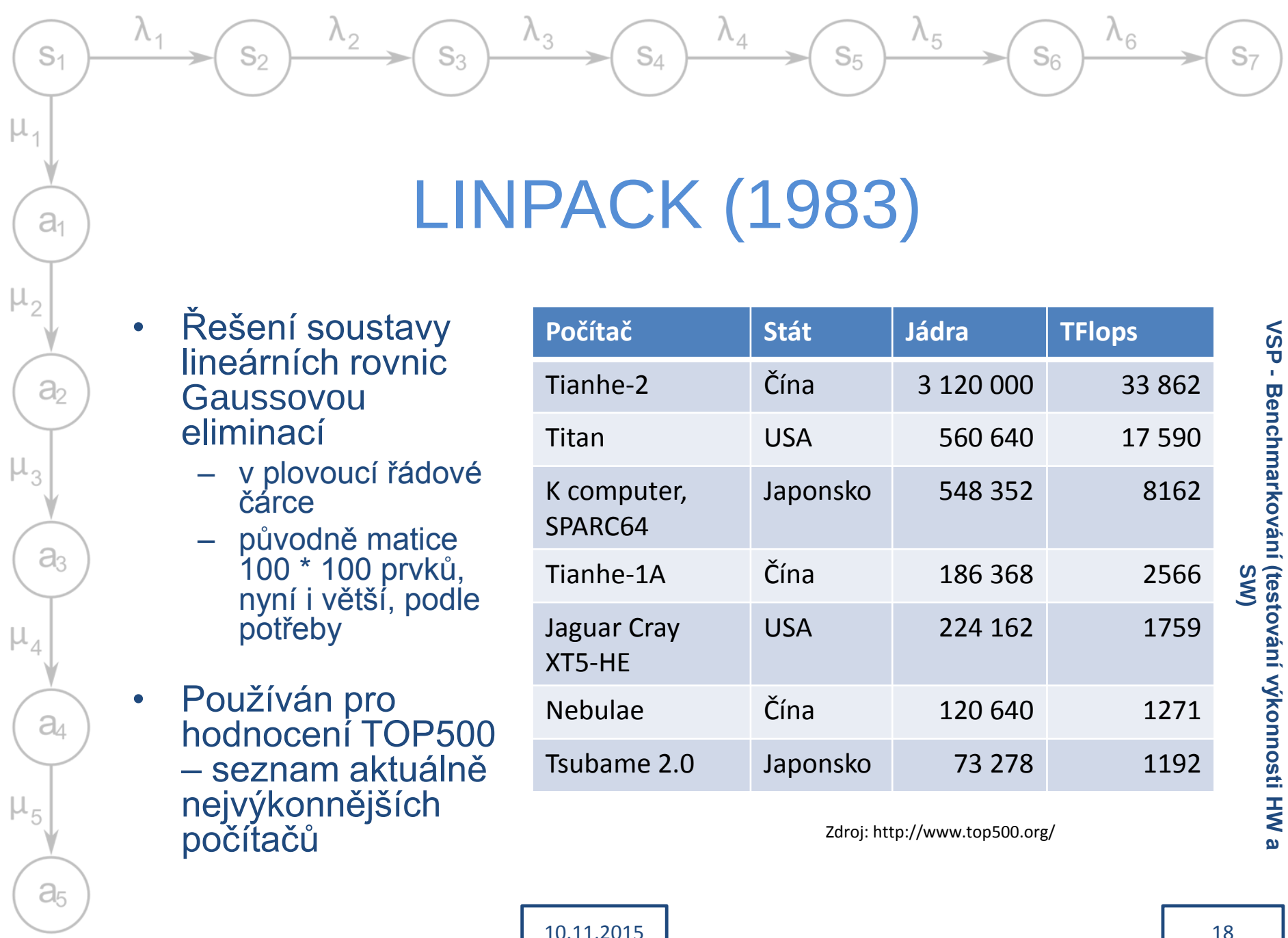


CoreMark (2009)

- Moderní náhrada za Dhrystone
  - Zpracování seznamů
  - Maticové operace
  - Konečný automat
  - Výpočer CRC
- Vlastní score

CPU	Frekvence (MHz)	Vlákna	CoreMark
Core2 Duo	1596	2	5115.18
Pentium M 760	2000	1	5298.00
Core i7 950 3060	3060	8	48343.68
Athlon 64 X2 QL-65 2100	2100	1	8918.62
Phenom II X6 1090T 3200	3200	6	73233.25

Zdroj: http://www.coremark.org/benchmark/

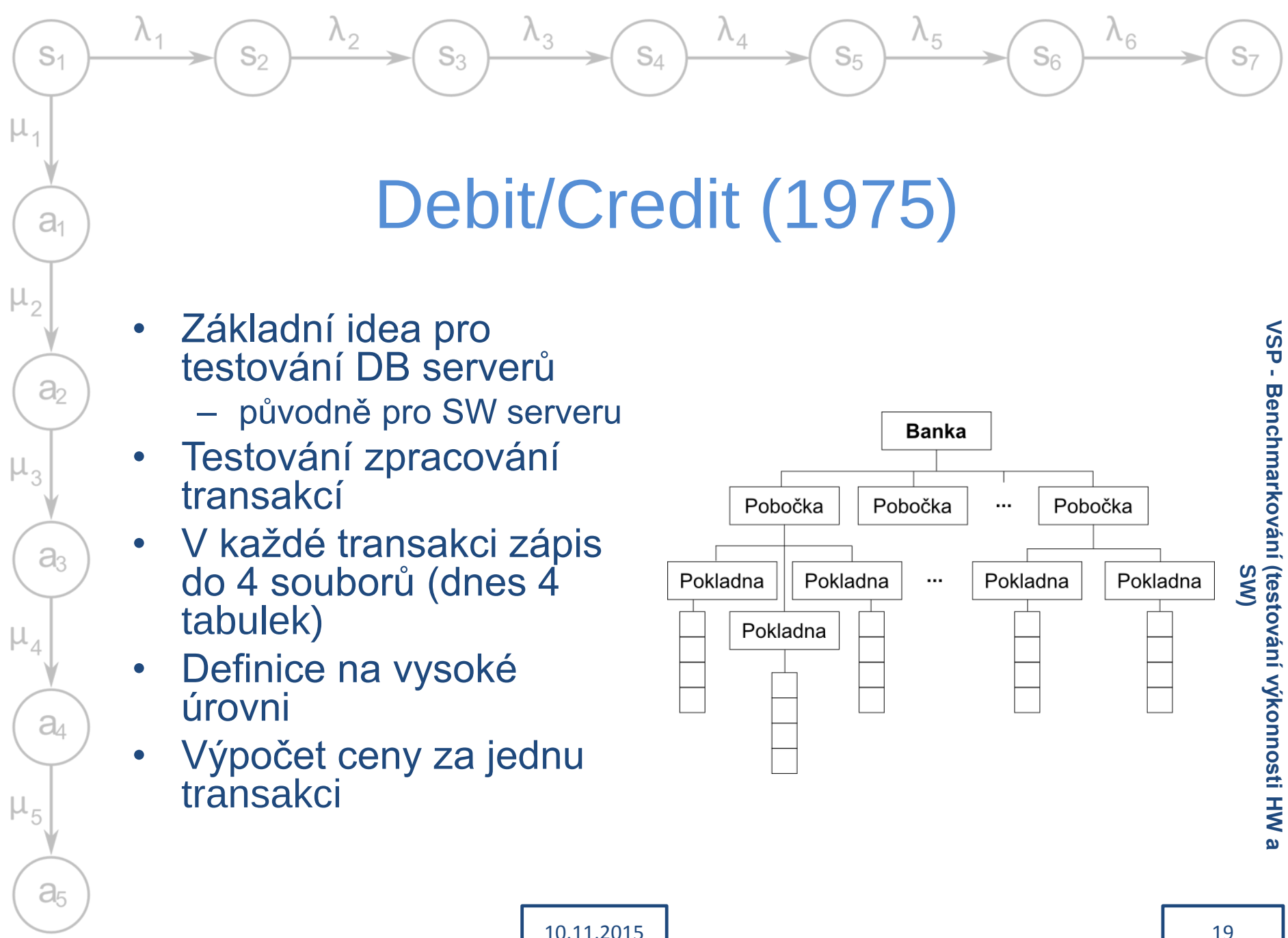


LINPACK (1983)

- Řešení soustavy lineárních rovnic Gaussovou eliminací
  - v plovoucí řádové čárce
  - původně matice 100 * 100 prvků, nyní i větší, podle potřeby
- Používán pro hodnocení TOP500 – seznam aktuálně nejvýkonnějších počítačů

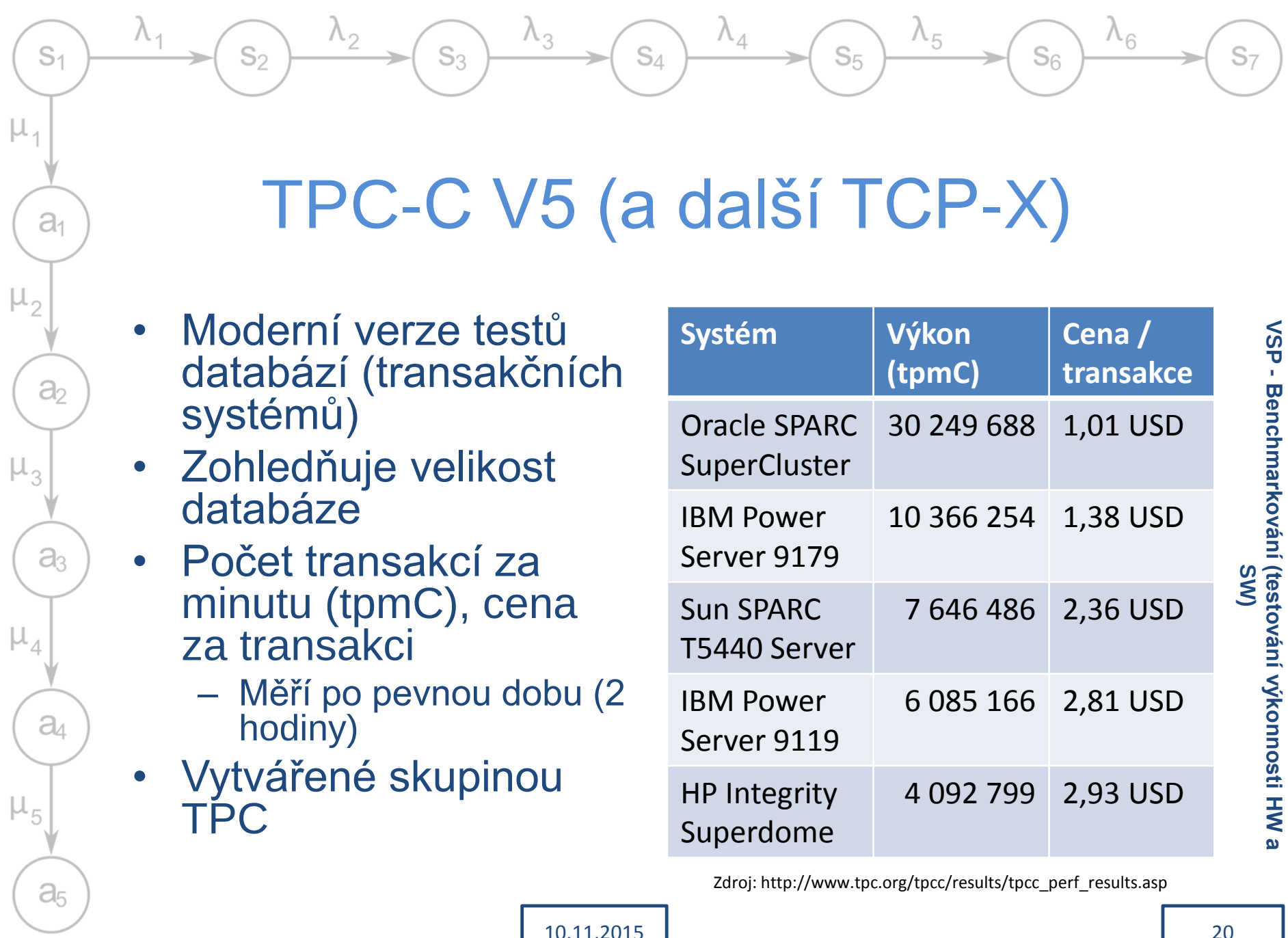
Počítač	Stát	Jádra	TFlops
Tianhe-2	Čína	3 120 000	33 862
Titan	USA	560 640	17 590
K computer, SPARC64	Japonsko	548 352	8162
Tianhe-1A	Čína	186 368	2566
Jaguar Cray XT5-HE	USA	224 162	1759
Nebulae	Čína	120 640	1271
Tsubame 2.0	Japonsko	73 278	1192

Zdroj: http://www.top500.org/



Debit/Credit (1975)

- Základní idea pro testování DB serverů
  - původně pro SW serveru
- Testování zpracování transakcí
- V každé transakci zápis do 4 souborů (dnes 4 tabulek)
- Definice na vysoké úrovni
- Výpočet ceny za jednu transakci

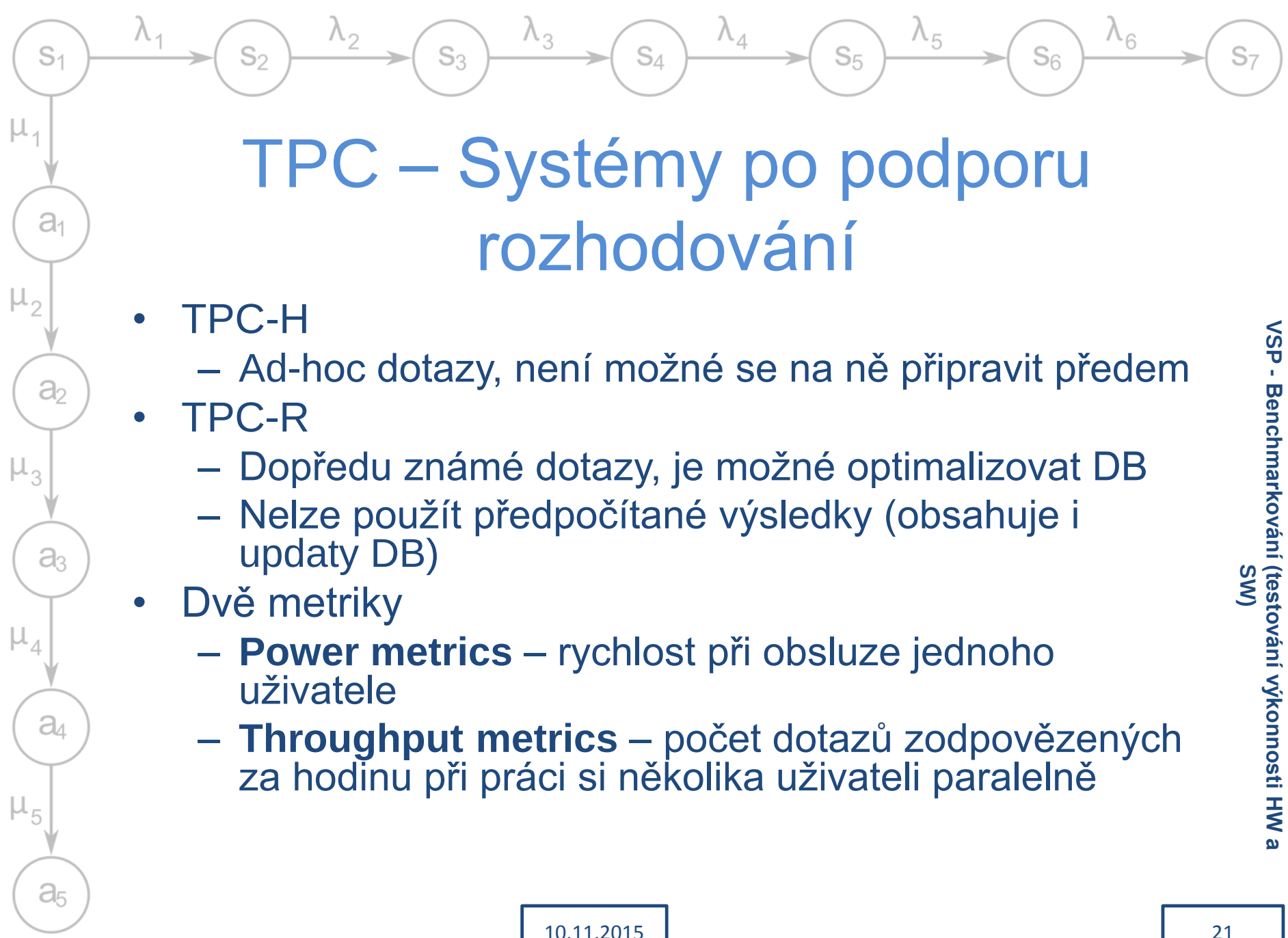


TPC-C V5 (a další TCP-X)

- Moderní verze testů databází (transakčních systémů)
- Zohledňuje velikost databáze
- Počet transakcí za minutu (tpmC), cena za transakci
  - Měří po pevnou dobu (2 hodiny)
- Vytvářené skupinou TPC

Systém	Výkon (tpmC)	Cena / transakce
Oracle SPARC SuperCluster	30 249 688	1,01 USD
IBM Power Server 9179	10 366 254	1,38 USD
Sun SPARC T5440 Server	7 646 486	2,36 USD
IBM Power Server 9119	6 085 166	2,81 USD
HP Integrity Superdome	4 092 799	2,93 USD

Zdroj: http://www.tpc.org/tpcc/results/tpcc_perf_results.asp



TPC – Systémy po podporu rozhodování

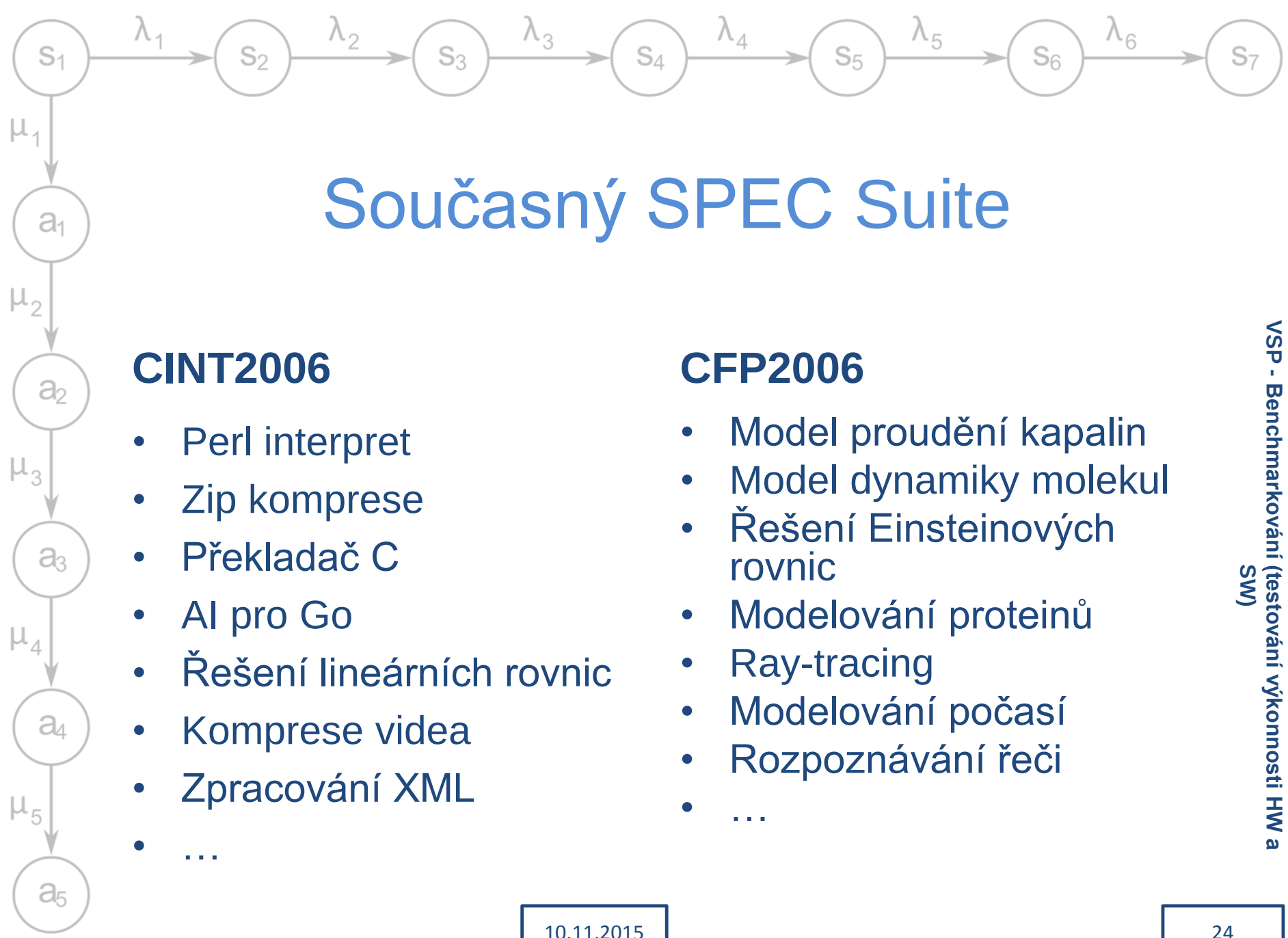
- TPC-H
  - Ad-hoc dotazy, není možné se na ně připravit předem
- TPC-R
  - Dopředu známé dotazy, je možné optimalizovat DB
  - Nelze použít předpočítané výsledky (obsahuje i updaty DB)
- Dvě metriky
  - **Power metrics** – rychlost při obsluze jednoho uživatele
  - **Throughput metrics** – počet dotazů zodpovězených za hodinu při práci si několika uživateli paralelně

TPC – další benchmarky

- TPC-W
  - Simulace aktivity obchodního serveru s webovým rozhraním
  - Metrikou počet obsloužených požadavků za sekundu
- TPC-E
  - Simulace firmy obchodující s cennými papíry
  - Komunikace se zákazníky a s burzou
  - Metrikou počet požadavků za vteřinu

SPEC Suite

- Sada různých reálných programů poskytnutých jako benchmarky
- Testování různých komponent / vlastností počítače
  - CPU, GPU, RAM
  - MPI (paralelismus)
  - Webové servery
  - Spotřeba
  - …



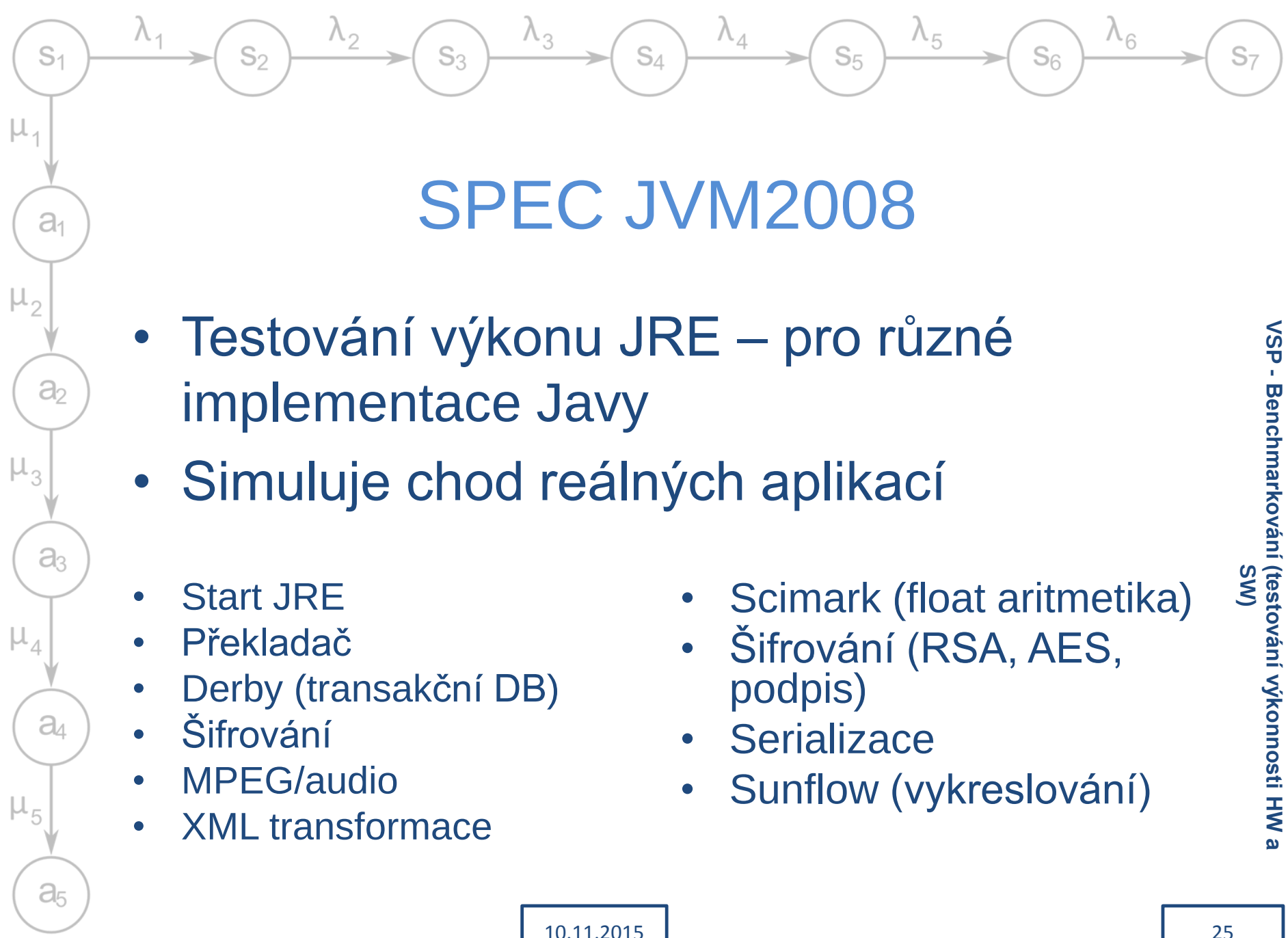
Současný SPEC Suite

CINT2006

- Perl interpret
- Zip komprese
- Překladač C
- AI pro Go
- Řešení lineárních rovnic
- Komprese videa
- Zpracování XML
- …

CFP2006

- Model proudění kapalin
- Model dynamiky molekul
- Řešení Einsteinových rovnic
- Modelování proteinů
- Ray-tracing
- Modelování počasí
- Rozpoznávání řeči
- …

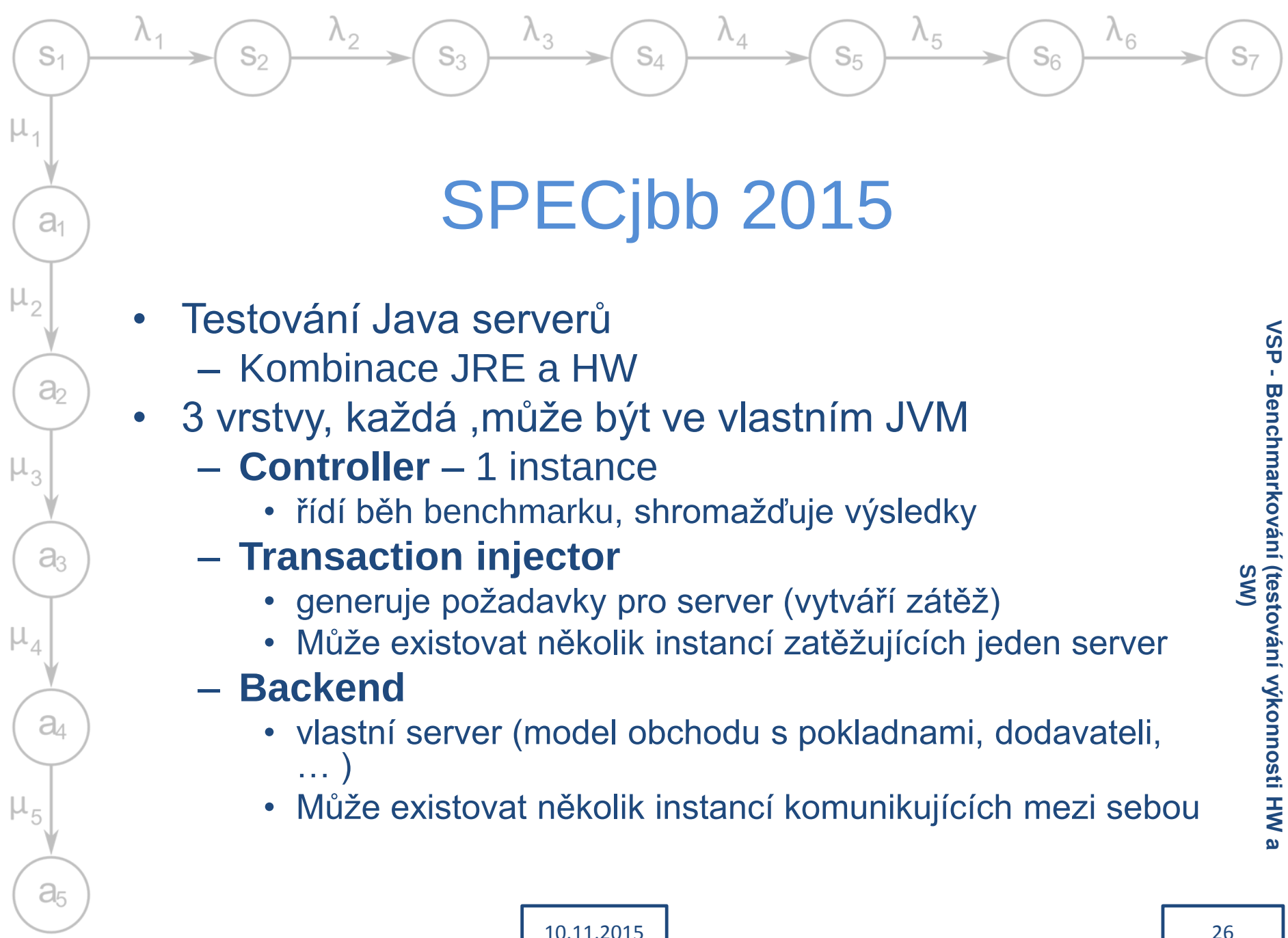


SPEC JVM2008

- Testování výkonu JRE – pro různé implementace Javy
- Simuluje chod reálných aplikací

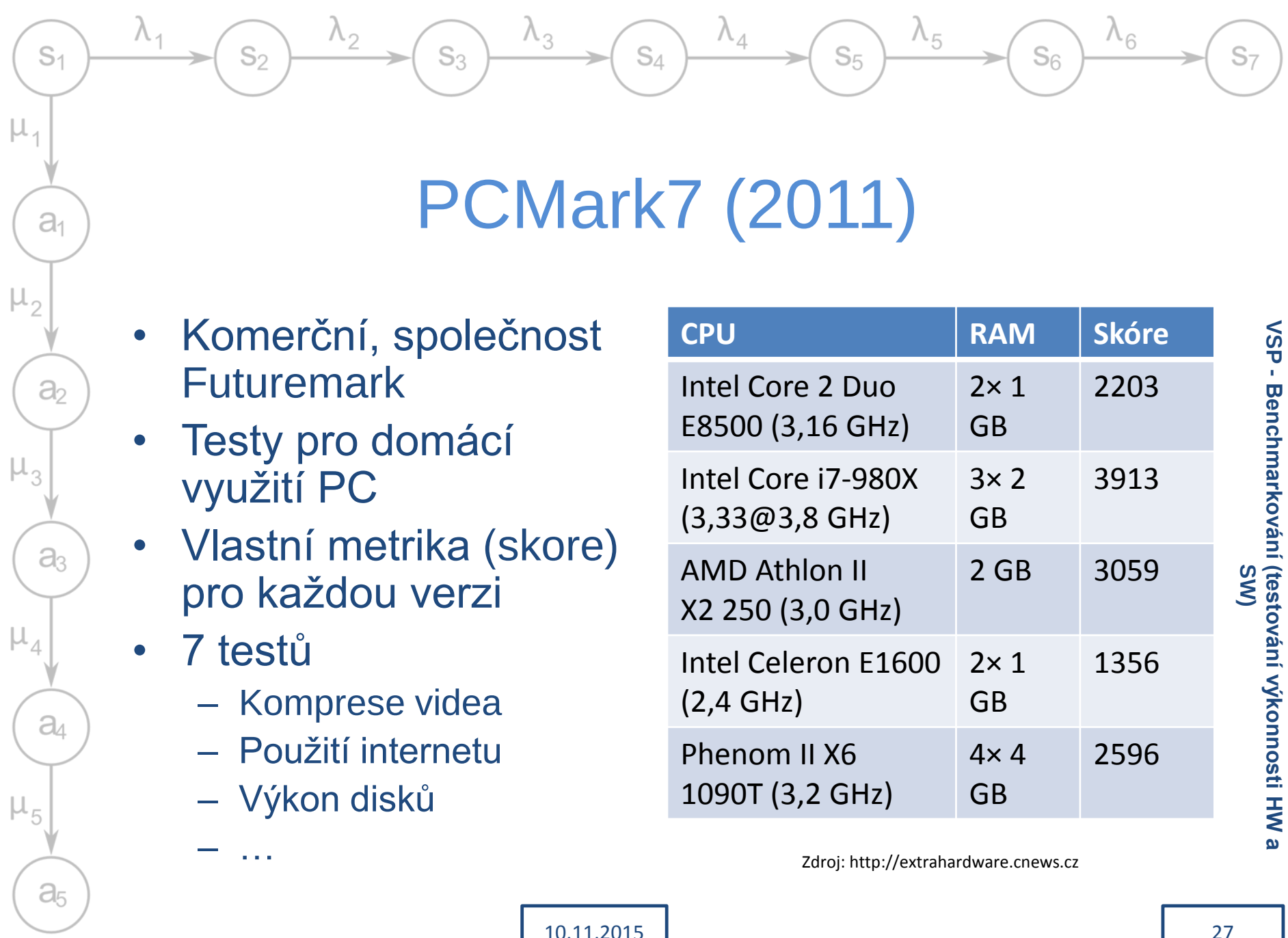
- Start JRE
- Překladač
- Derby (transakční DB)
- Šifrování
- MPEG/audio
- XML transformace

- Scimark (float aritmetika)
- Šifrování (RSA, AES, podpis)
- Serializace
- Sunflow (vykreslování)



SPECjbb 2015

- Testování Java serverů
  - Kombinace JRE a HW
- 3 vrstvy, každá ,může být ve vlastním JVM
  - **Controller** – 1 instance
    - řídí běh benchmarku, shromažďuje výsledky
  - **Transaction injector**
    - generuje požadavky pro server (vytváří zátěž)
    - Může existovat několik instancí zatěžujících jeden server
  - **Backend**
    - vlastní server (model obchodu s pokladnami, dodavateli, …)
    - Může existovat několik instancí komunikujících mezi sebou



PCMark7 (2011)

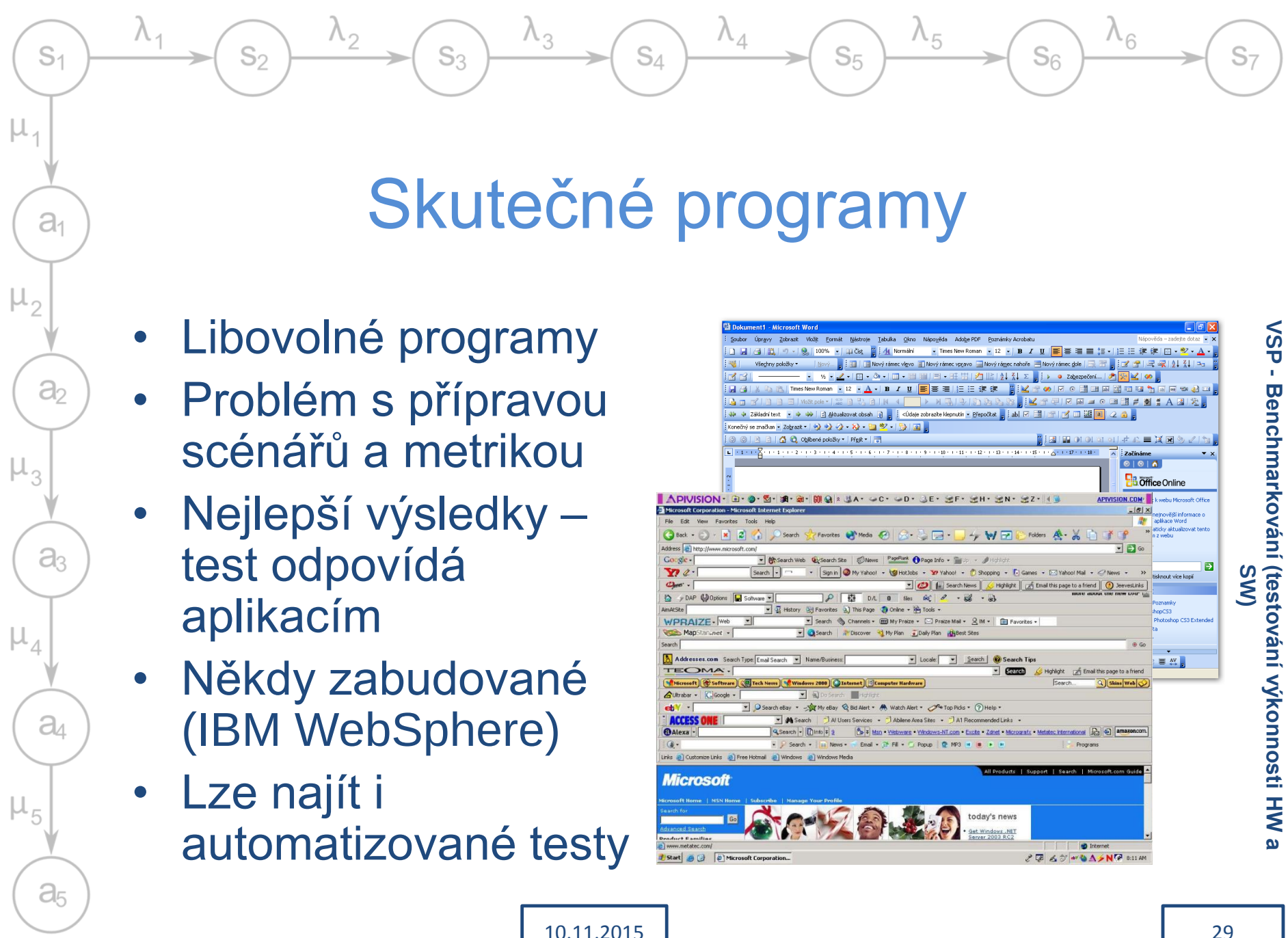
- Komerční, společnost Futuremark
- Testy pro domácí využití PC
- Vlastní metrika (skore) pro každou verzi
- 7 testů
  - Komprese videa
  - Použití internetu
  - Výkon disků
  - …

CPU	RAM	Skóre
Intel Core 2 Duo E8500 (3,16 GHz)	2× 1 GB	2203
Intel Core i7-980X (3,33@3,8 GHz)	3× 2 GB	3913
AMD Athlon II X2 250 (3,0 GHz)	2 GB	3059
Intel Celeron E1600 (2,4 GHz)	2× 1 GB	1356
Phenom II X6 1090T (3,2 GHz)	4× 4 GB	2596

Zdroj: http://extrahardware.cnews.cz

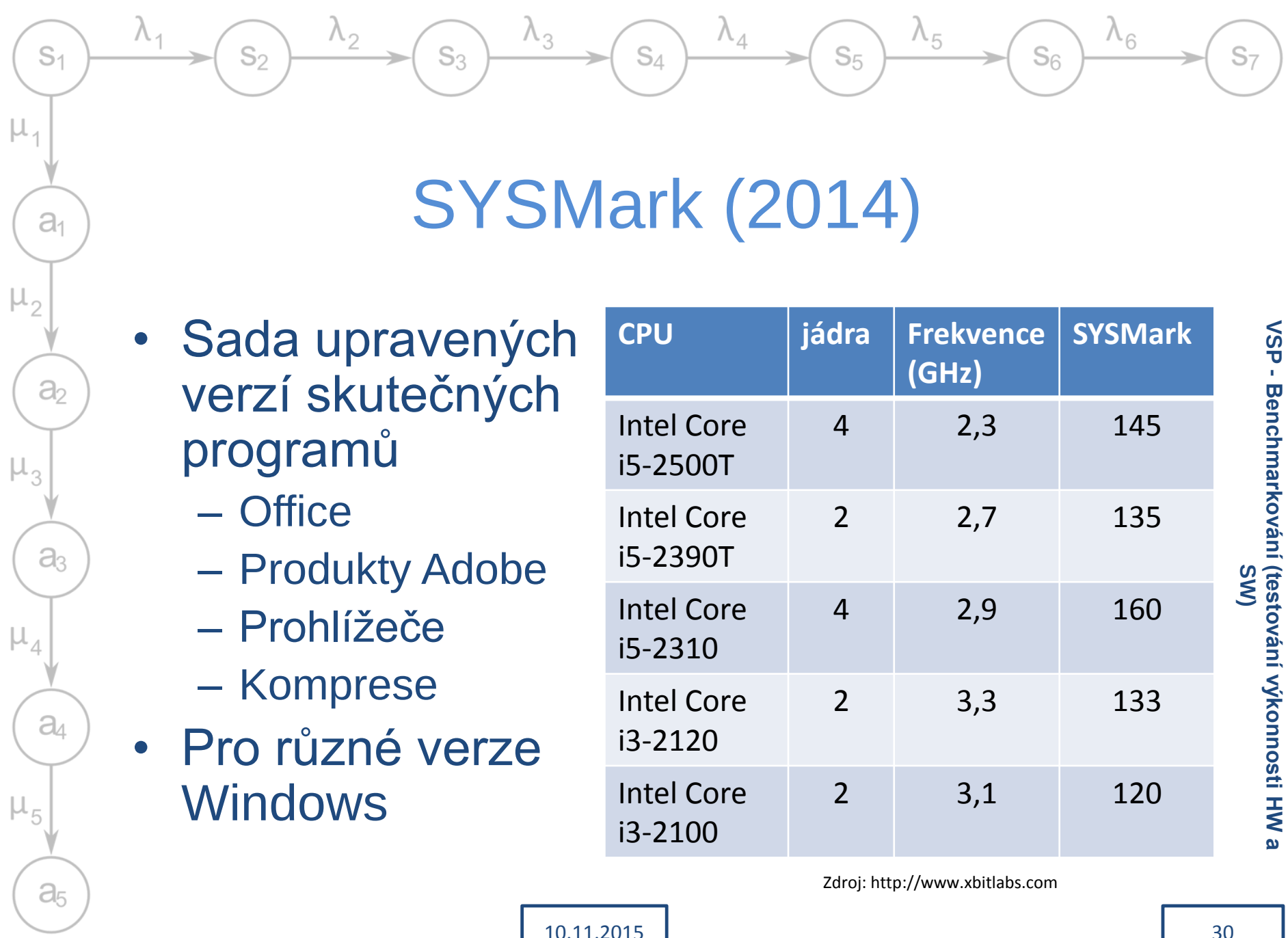
3DMark11 (2010)

- Komerční, společnost Futuremark
- Testování výkonu grafické karty a cpu, zaměřené na PC hry
- Různé verze podle verze DirectX
- Vlastní metrika



Skutečné programy

- Libovolné programy
- Problém s přípravou scénářů a metrikou
- Nejlepší výsledky – test odpovídá aplikacím
- Někdy zabudované (IBM WebSphere)
- Lze najít i automatizované testy

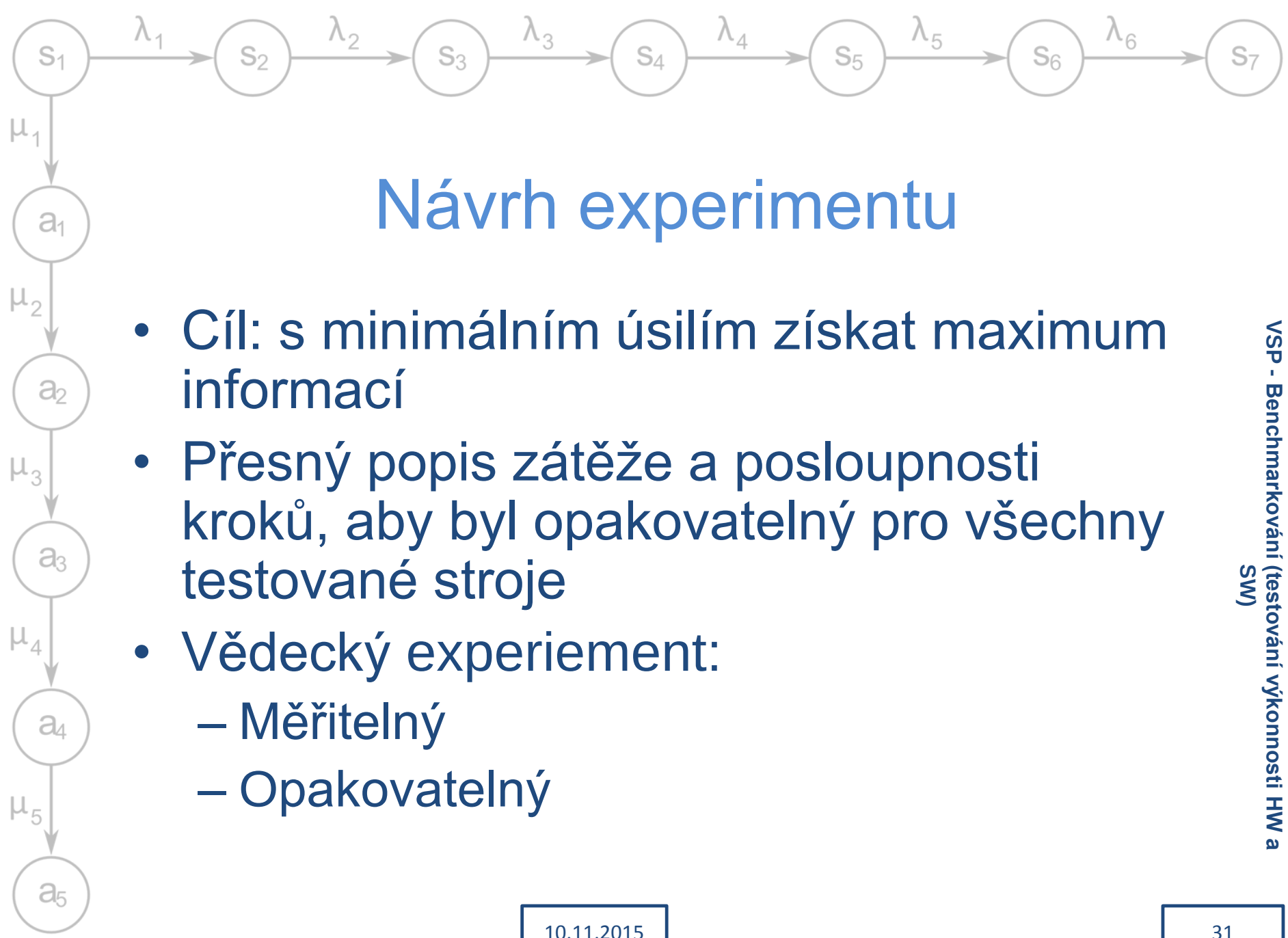


SYSMark (2014)

- Sada upravených verzí skutečných programů
  - Office
  - Produkty Adobe
  - Prohlížeče
  - Komprese
- Pro různé verze Windows

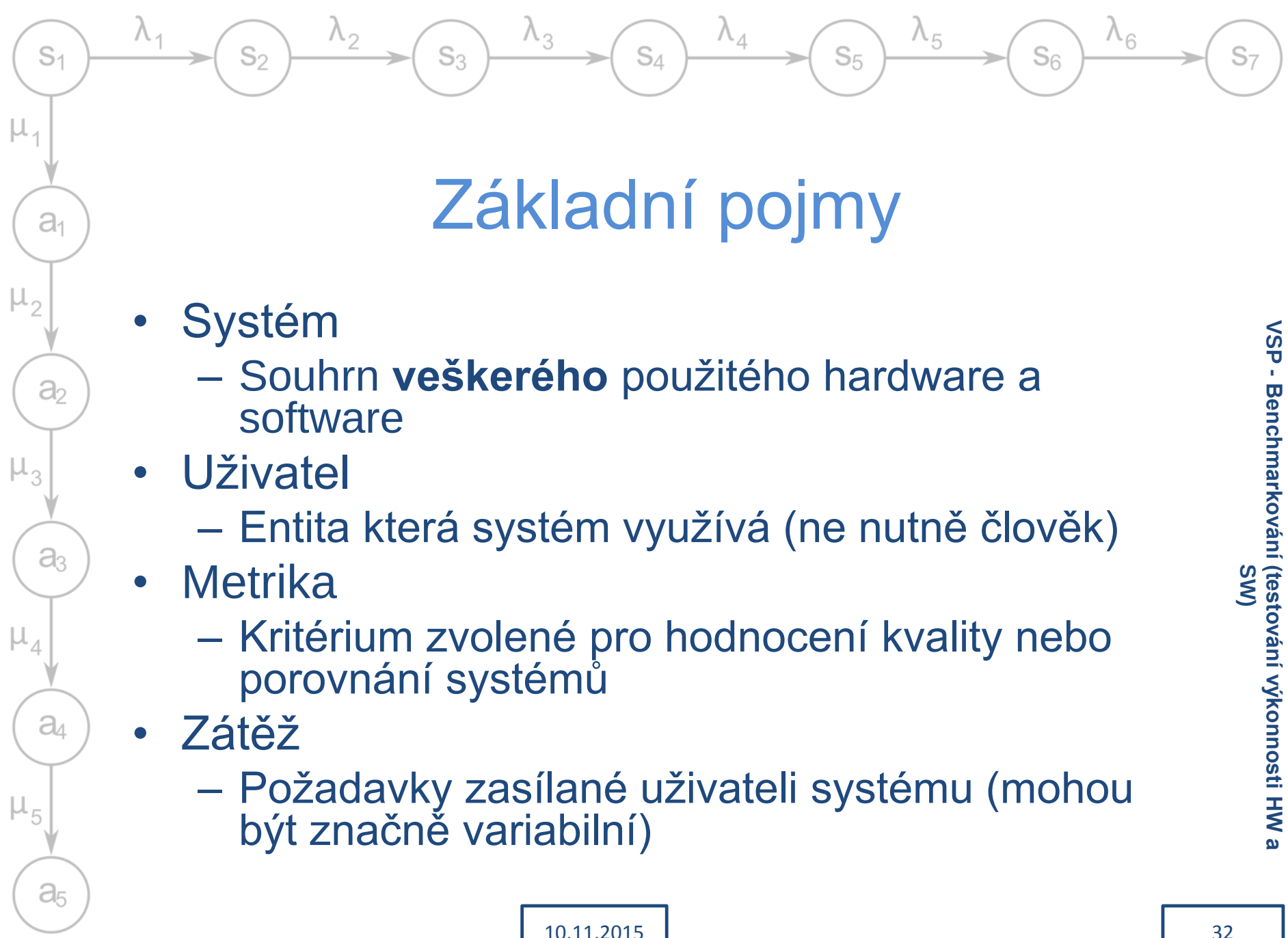
CPU	jádra	Frekvence (GHz)	SYSMark
Intel Core i5-2500T	4	2,3	145
Intel Core i5-2390T	2	2,7	135
Intel Core i5-2310	4	2,9	160
Intel Core i3-2120	2	3,3	133
Intel Core i3-2100	2	3,1	120

Zdroj: http://www.xbitlabs.com



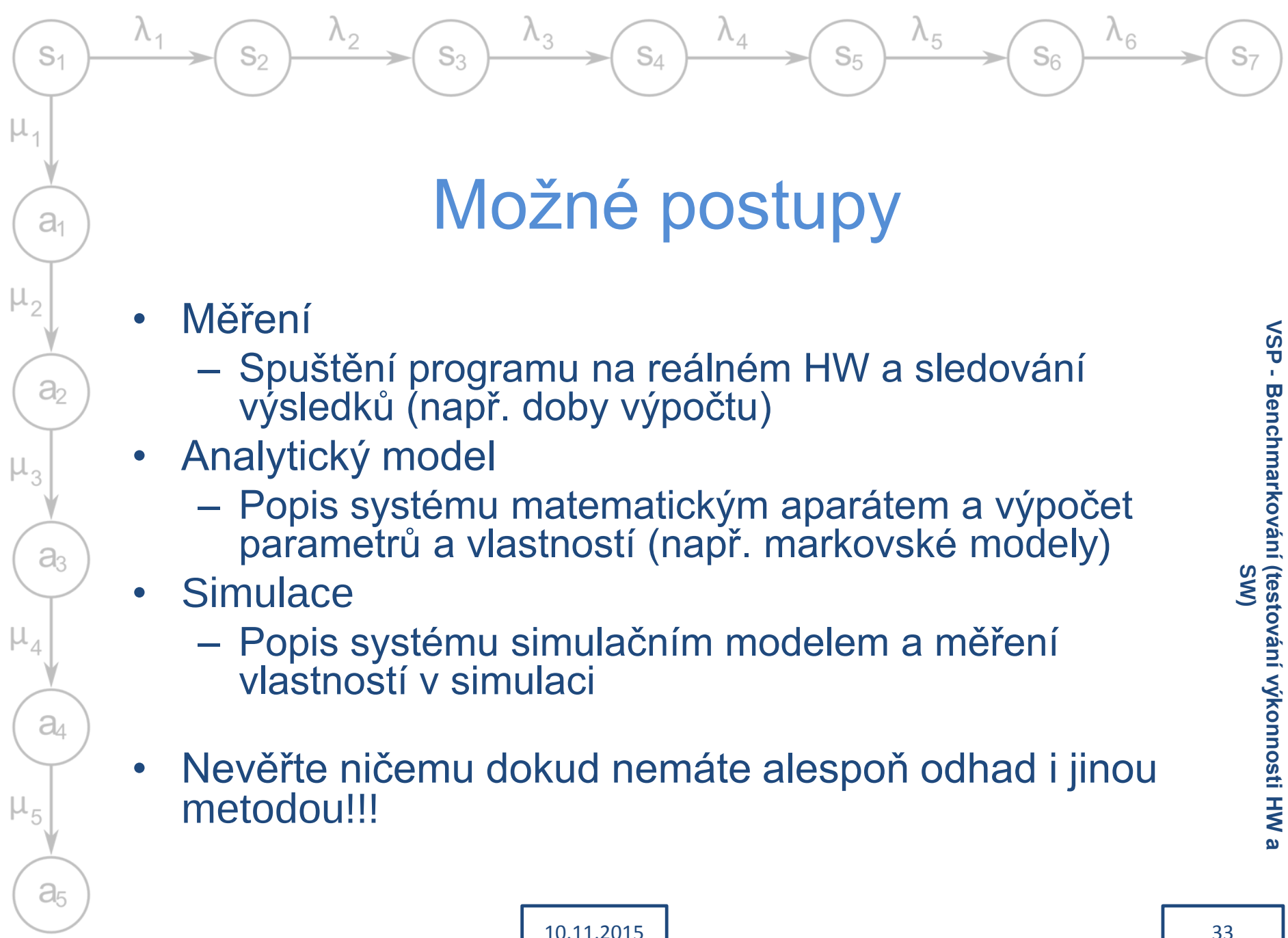
Návrh experimentu

- Cíl: s minimálním úsilím získat maximum informací
- Přesný popis zátěže a posloupnosti kroků, aby byl opakovatelný pro všechny testované stroje
- Vědecký experiement:
  - Měřitelný
  - Opakovatelný



Základní pojmy

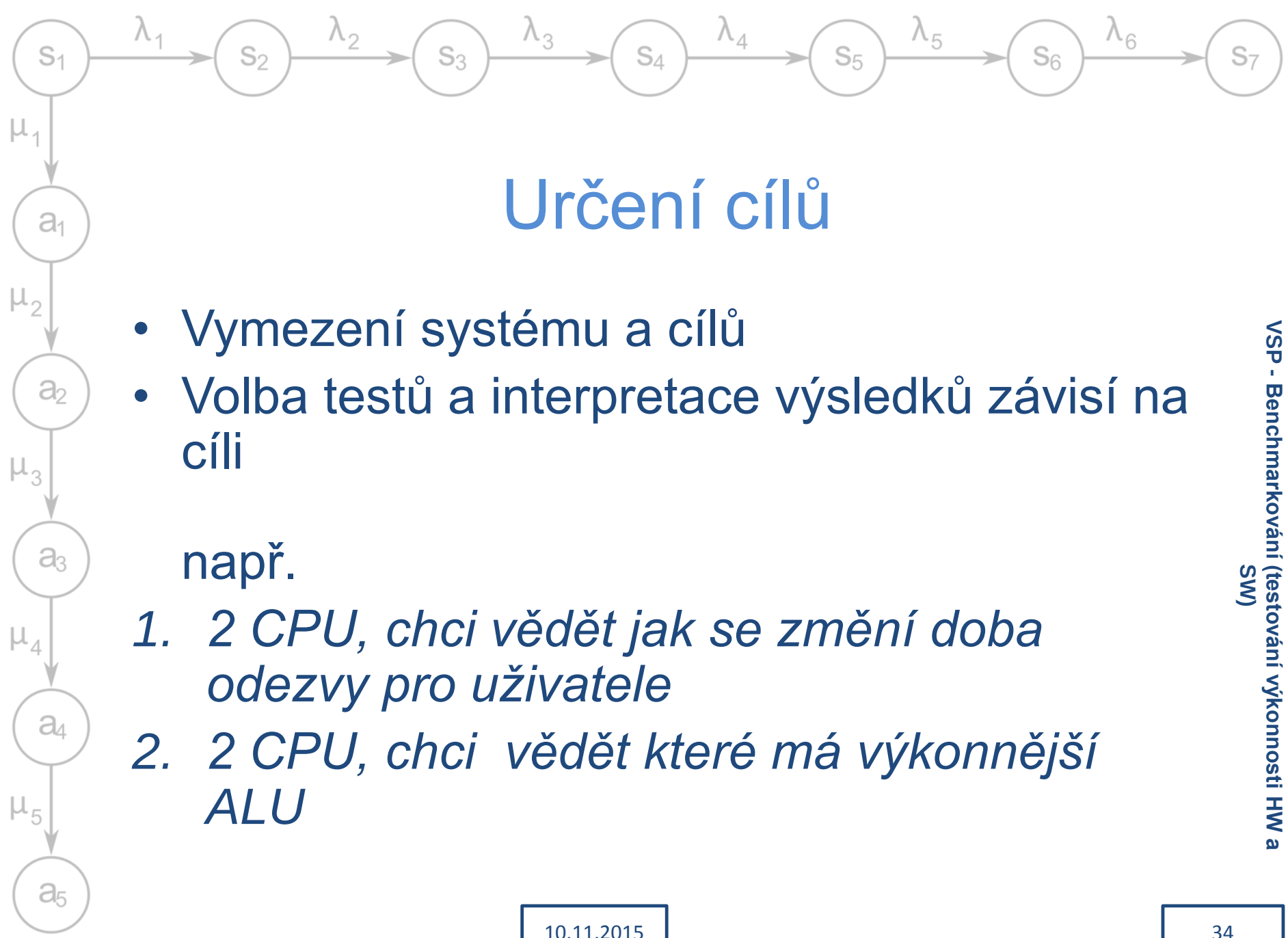
- Systém
  - Souhrn **veškerého** použitého hardware a software
- Uživatel
  - Entita která systém využívá (ne nutně člověk)
- Metrika
  - Kritérium zvolené pro hodnocení kvality nebo porovnání systémů
- Zátěž
  - Požadavky zasílané uživateli systému (mohou být značně variabilní)



Možné postupy

- Měření
  - Spuštění programu na reálném HW a sledování výsledků (např. doby výpočtu)
- Analytický model
  - Popis systému matematickým aparátem a výpočet parametrů a vlastností (např. markovské modely)
- Simulace
  - Popis systému simulačním modelem a měření vlastností v simulaci

- Nevěřte ničemu dokud nemáte alespoň odhad i jinou metodou!!!

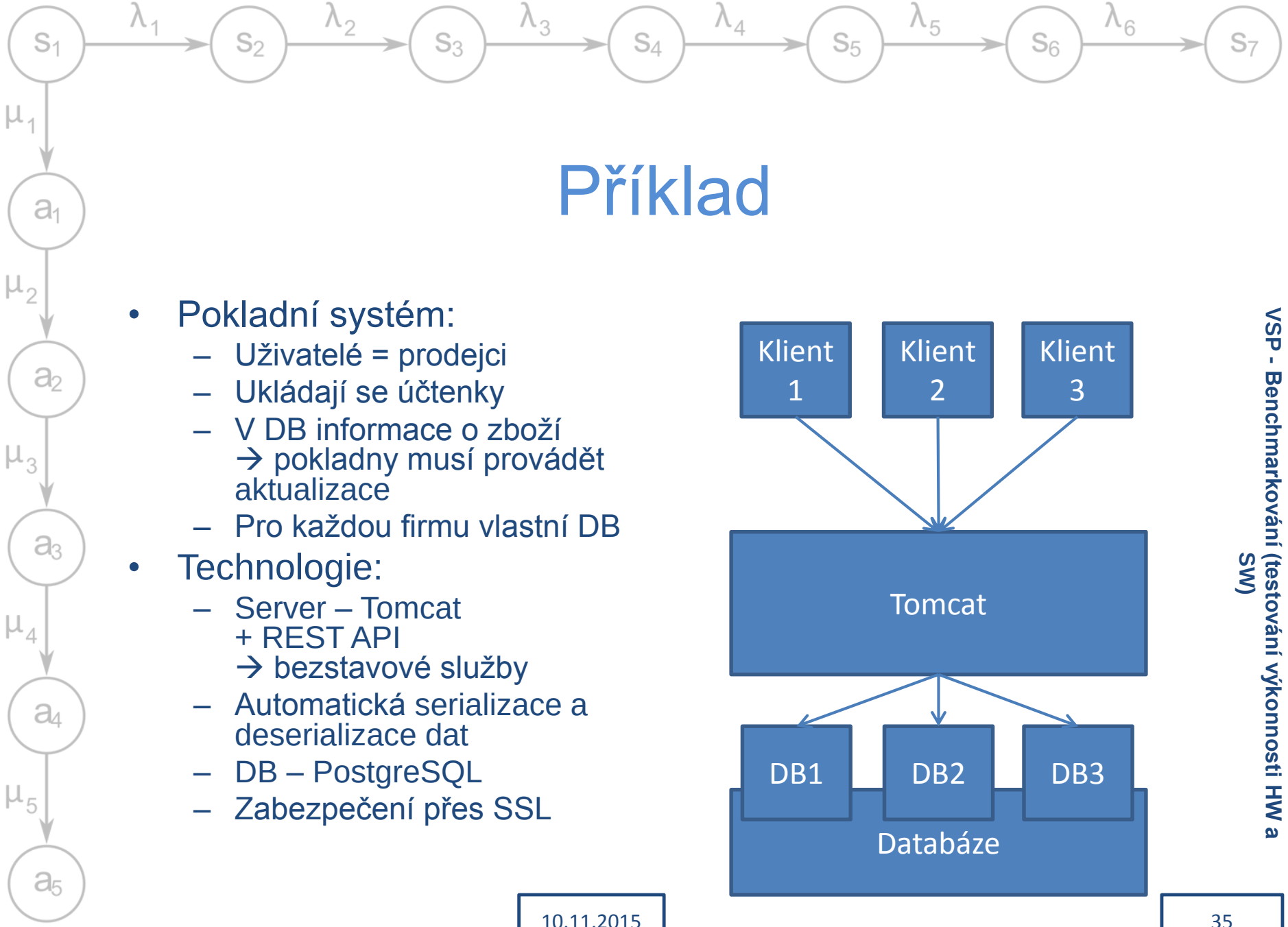


Určení cílů

- Vymezení systému a cílů
- Volba testů a interpretace výsledků závisí na cíli

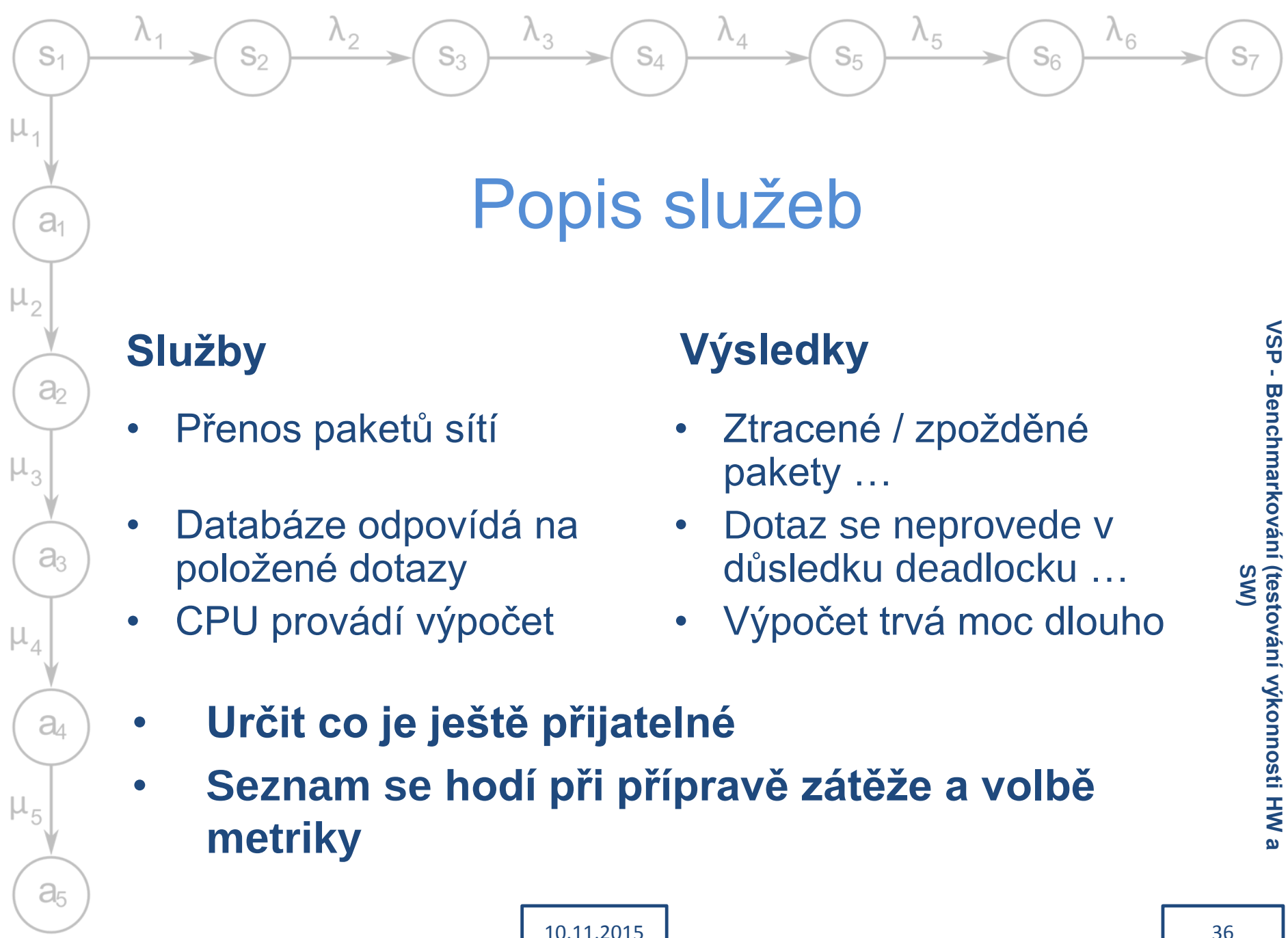
např.

1. *2 CPU, chci vědět jak se změní doba odezvy pro uživatele*
2. *2 CPU, chci vědět které má výkonnější ALU*



Příklad

- Pokladní systém:
  - Uživatelé = prodejci
  - Ukládají se účtenky
  - V DB informace o zboží
    → pokladny musí provádět aktualizace
  - Pro každou firmu vlastní DB
- Technologie:
  - Server – Tomcat
    + REST API
    → bezstavové služby
  - Automatická serializace a deserializace dat
  - DB – PostgreSQL
  - Zabezpečení přes SSL



Popis služeb

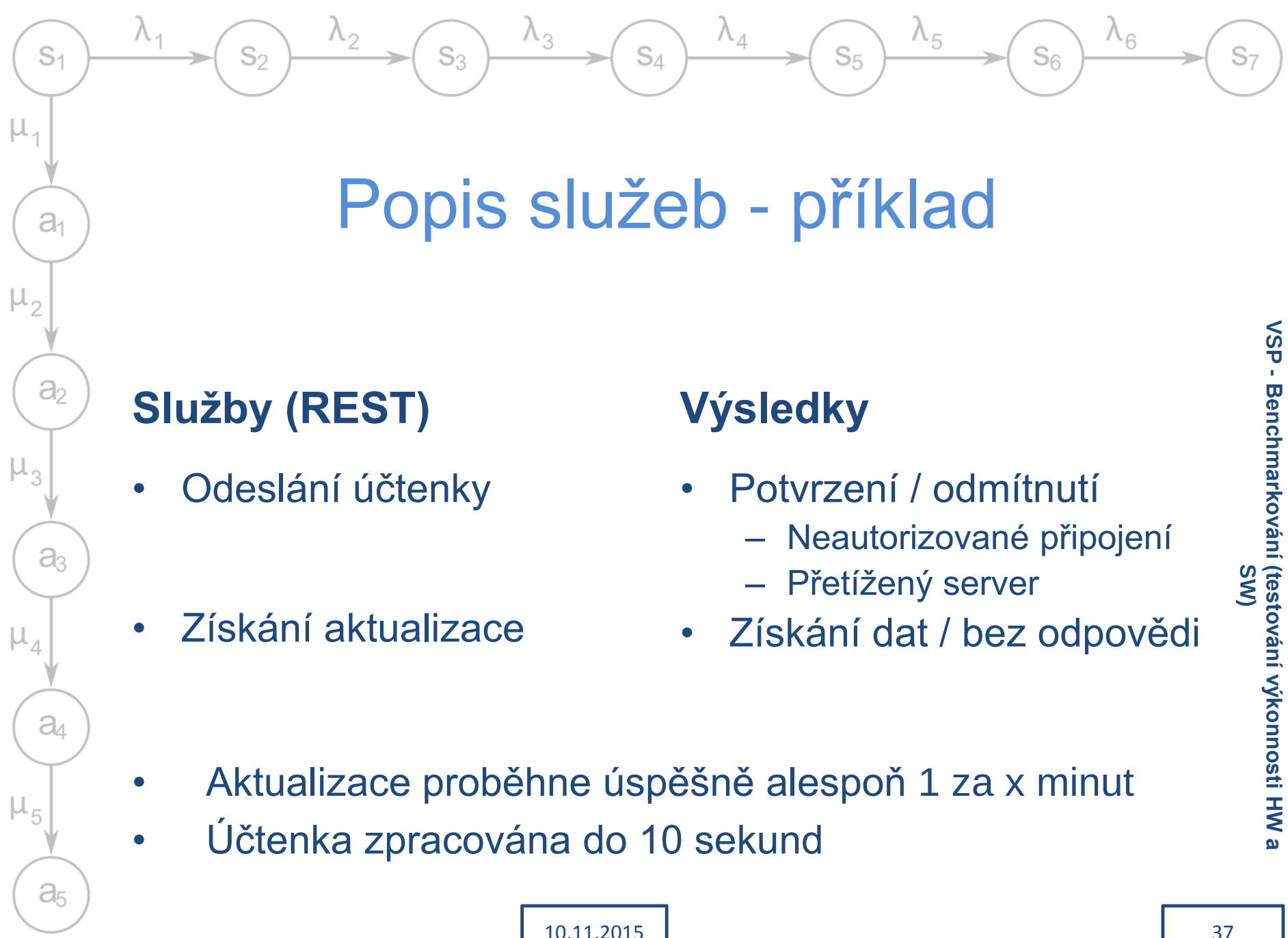
Služby

- Přenos paketů sítí
- Databáze odpovídá na položené dotazy
- CPU provádí výpočet

Výsledky

- Ztracené / zpožděné pakety …
- Dotaz se neprovede v důsledku deadlocku …
- Výpočet trvá moc dlouho

- **Určit co je ještě přijatelné**
- **Seznam se hodí při přípravě zátěže a volbě metriky**



Popis služeb - příklad

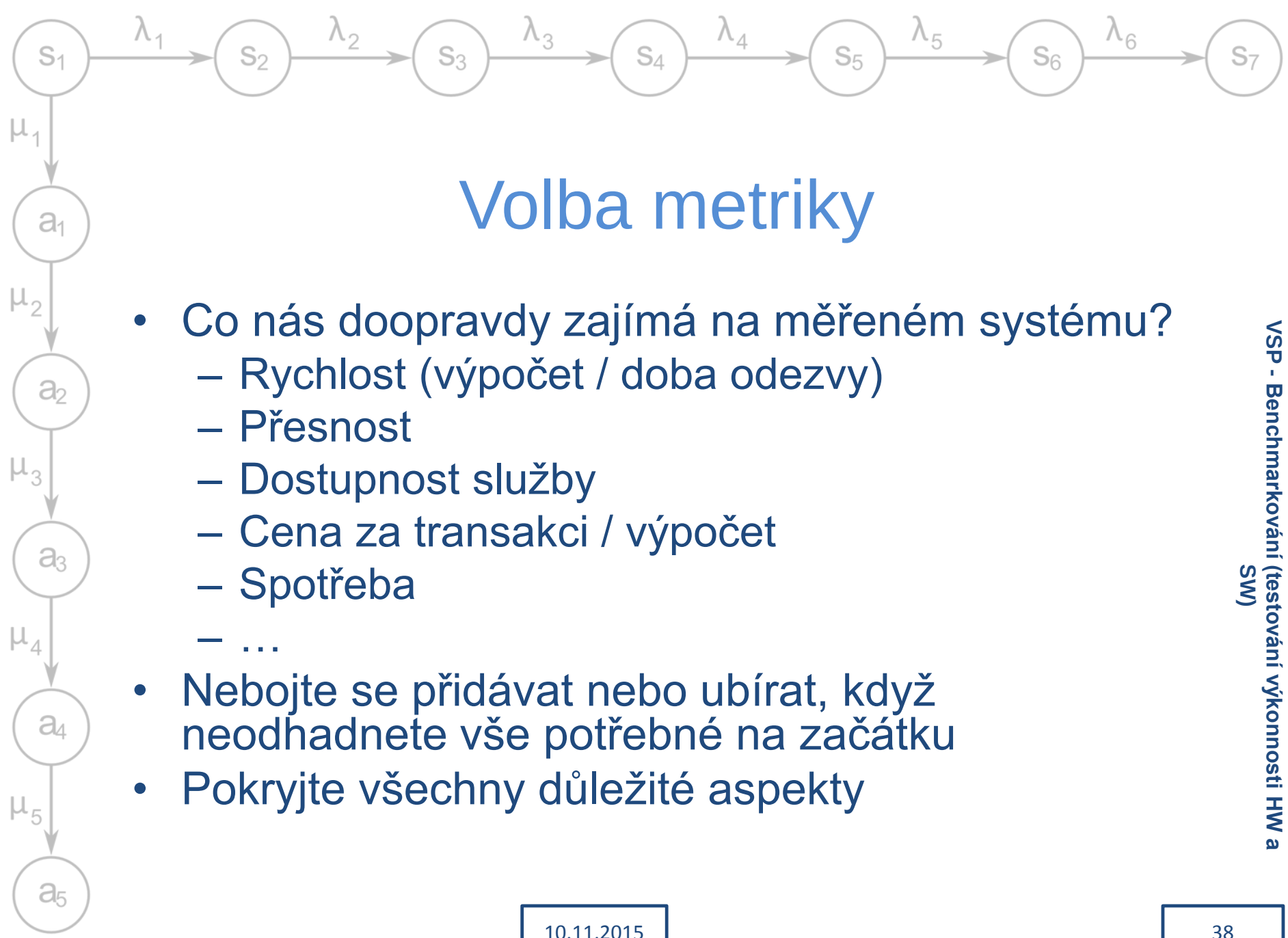
Služby (REST)

- Odeslání účtenky
- Získání aktualizace

Výsledky

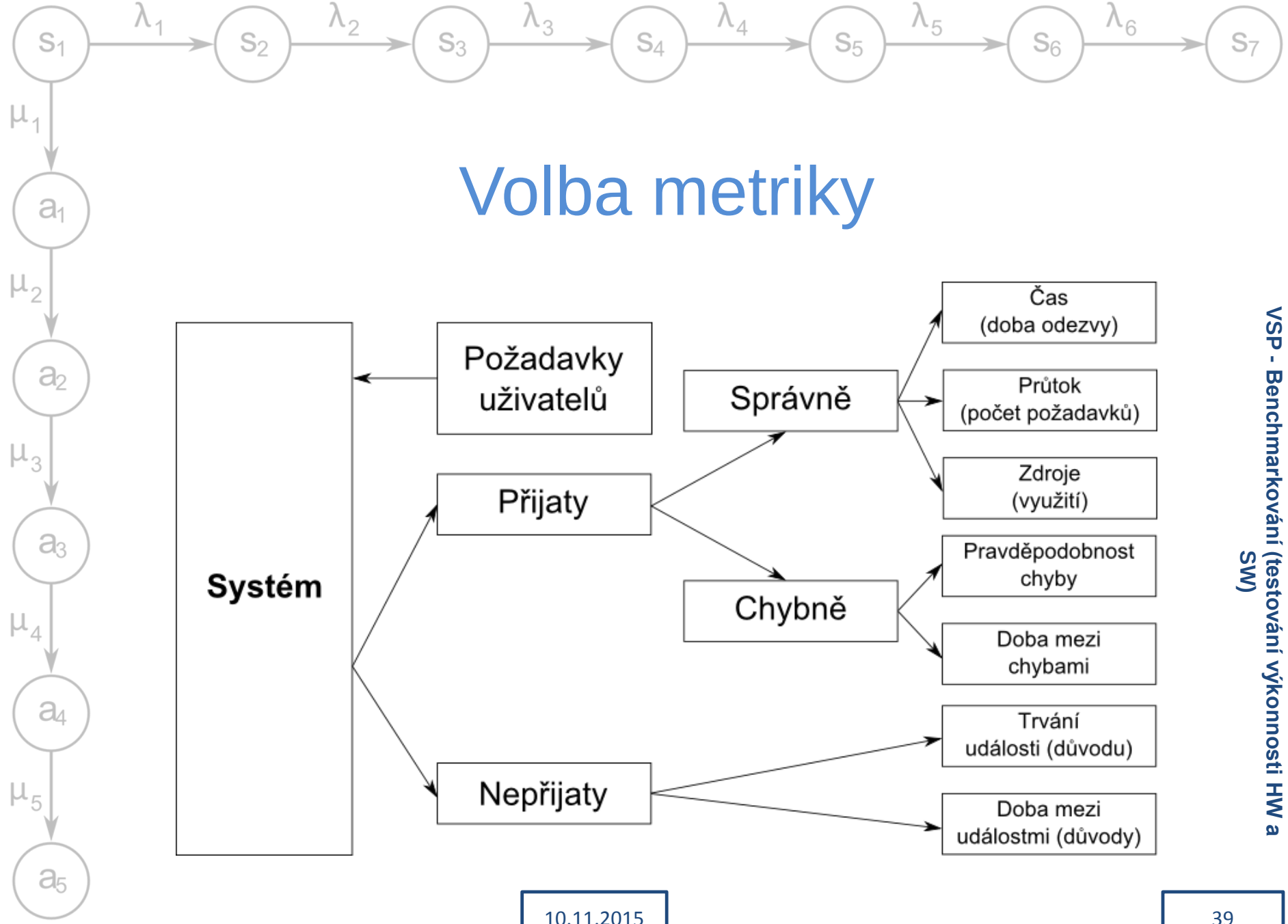
- Potvrzení / odmítnutí
  - Neautorizované připojení
  - Přetížený server
- Získání dat / bez odpovědi

- Aktualizace proběhne úspěšně alespoň 1 za x minut
- Účtenka zpracována do 10 sekund

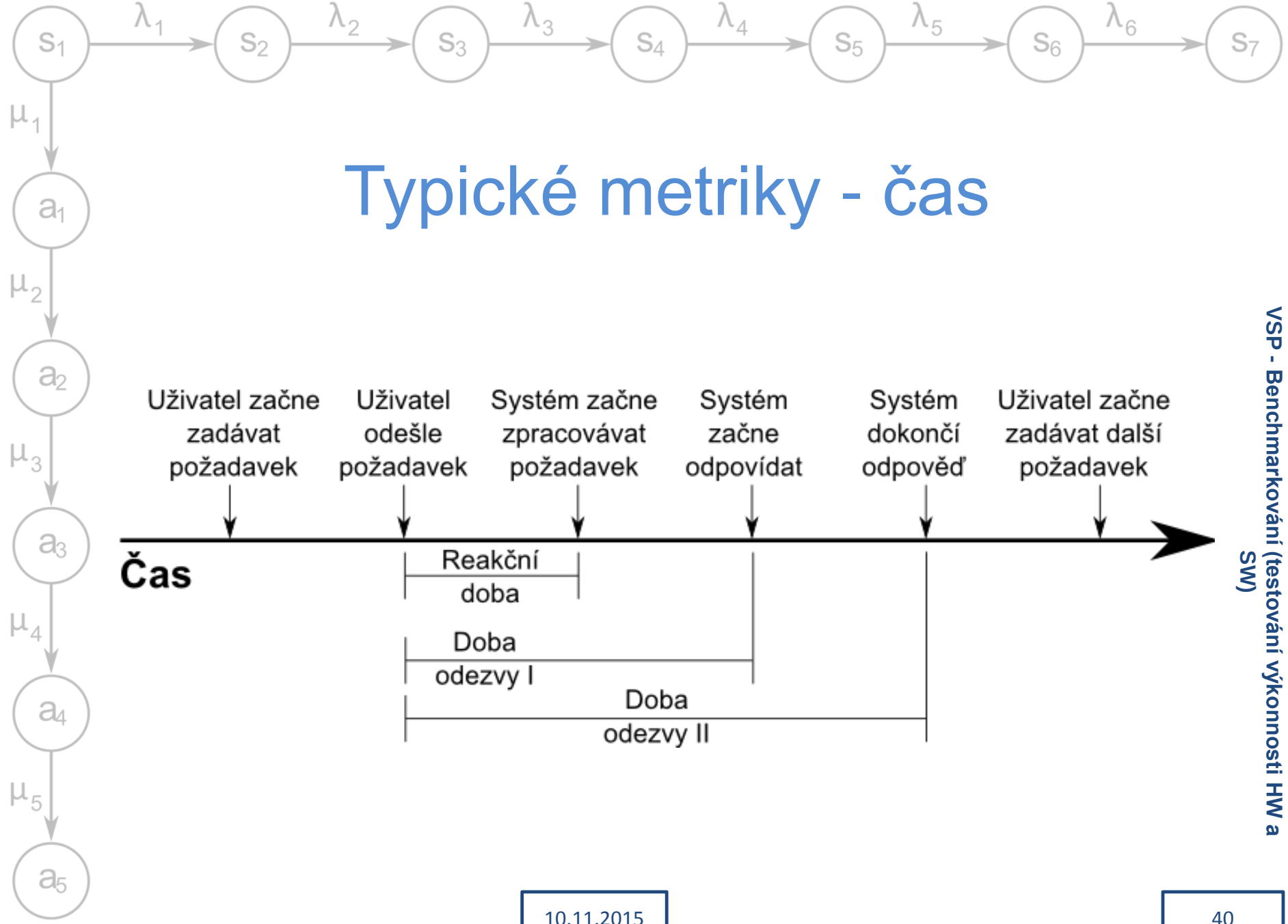


Volba metriky

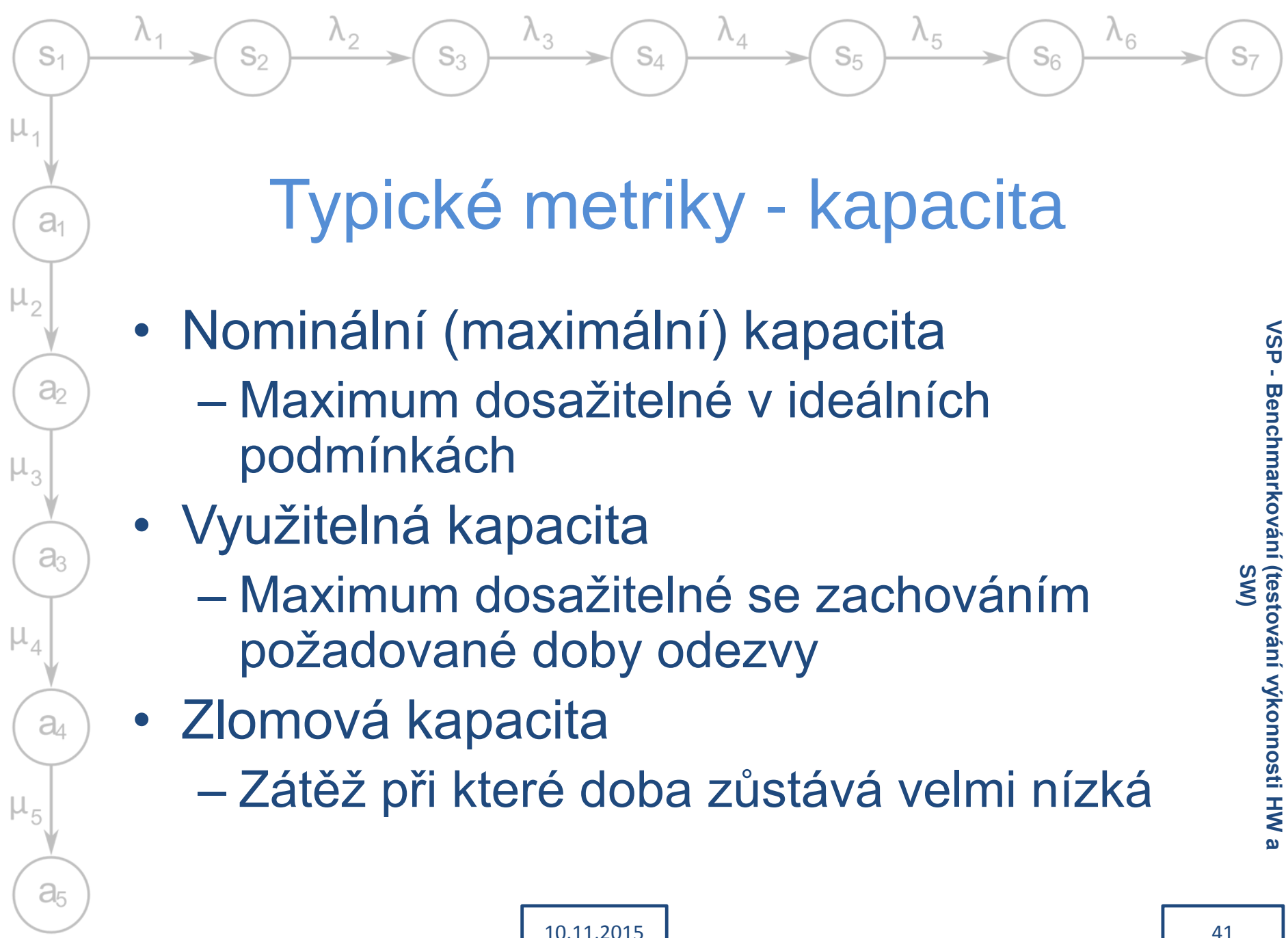
- Co nás doopravdy zajímá na měřeném systému?
  - Rychlost (výpočet / doba odezvy)
  - Přesnost
  - Dostupnost služby
  - Cena za transakci / výpočet
  - Spotřeba
  - …
- Nebojte se přidávat nebo ubírat, když neodhadnete vše potřebné na začátku
- Pokryjte všechny důležité aspekty



Volba metriky

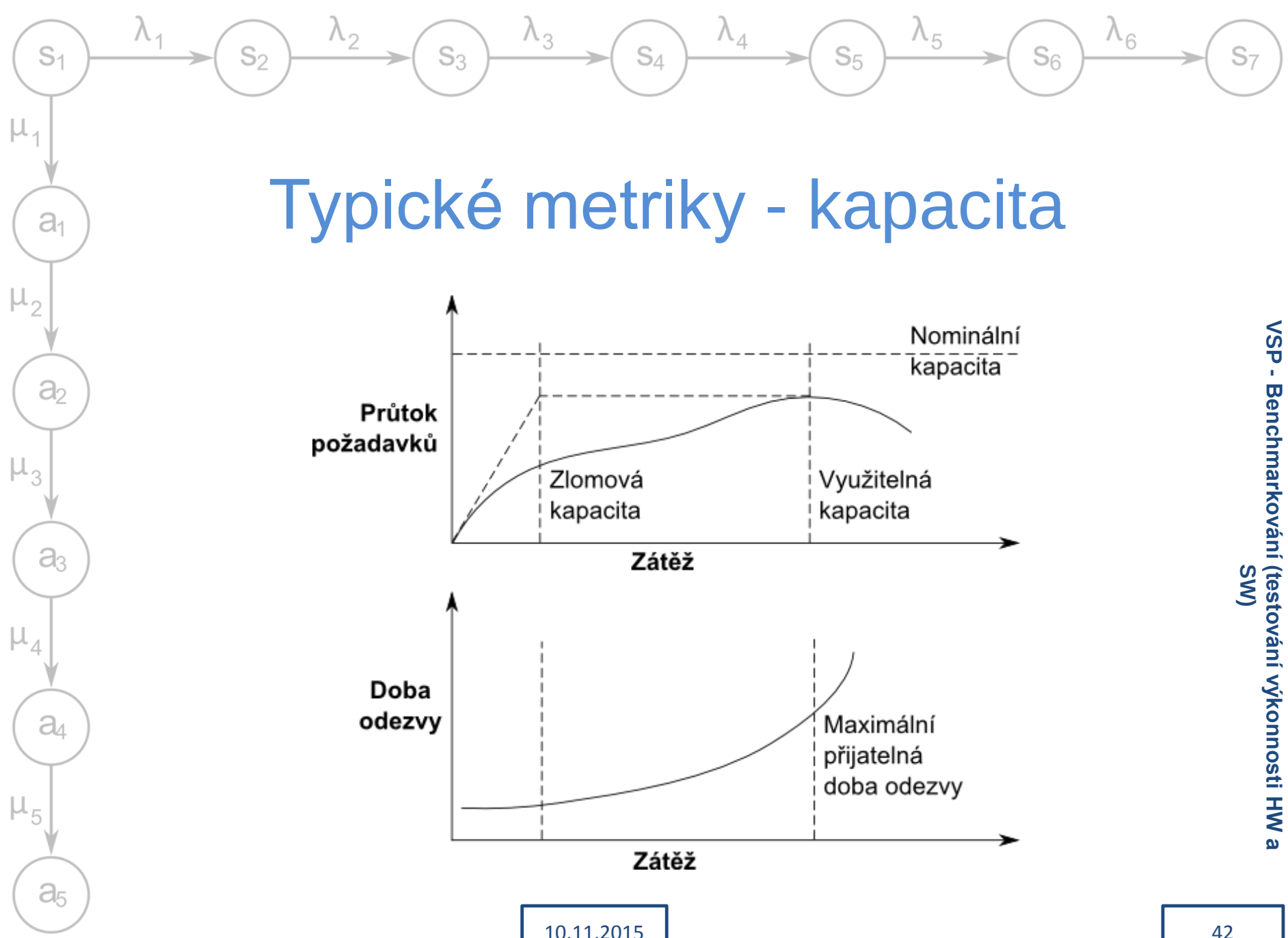


Typické metriky - čas

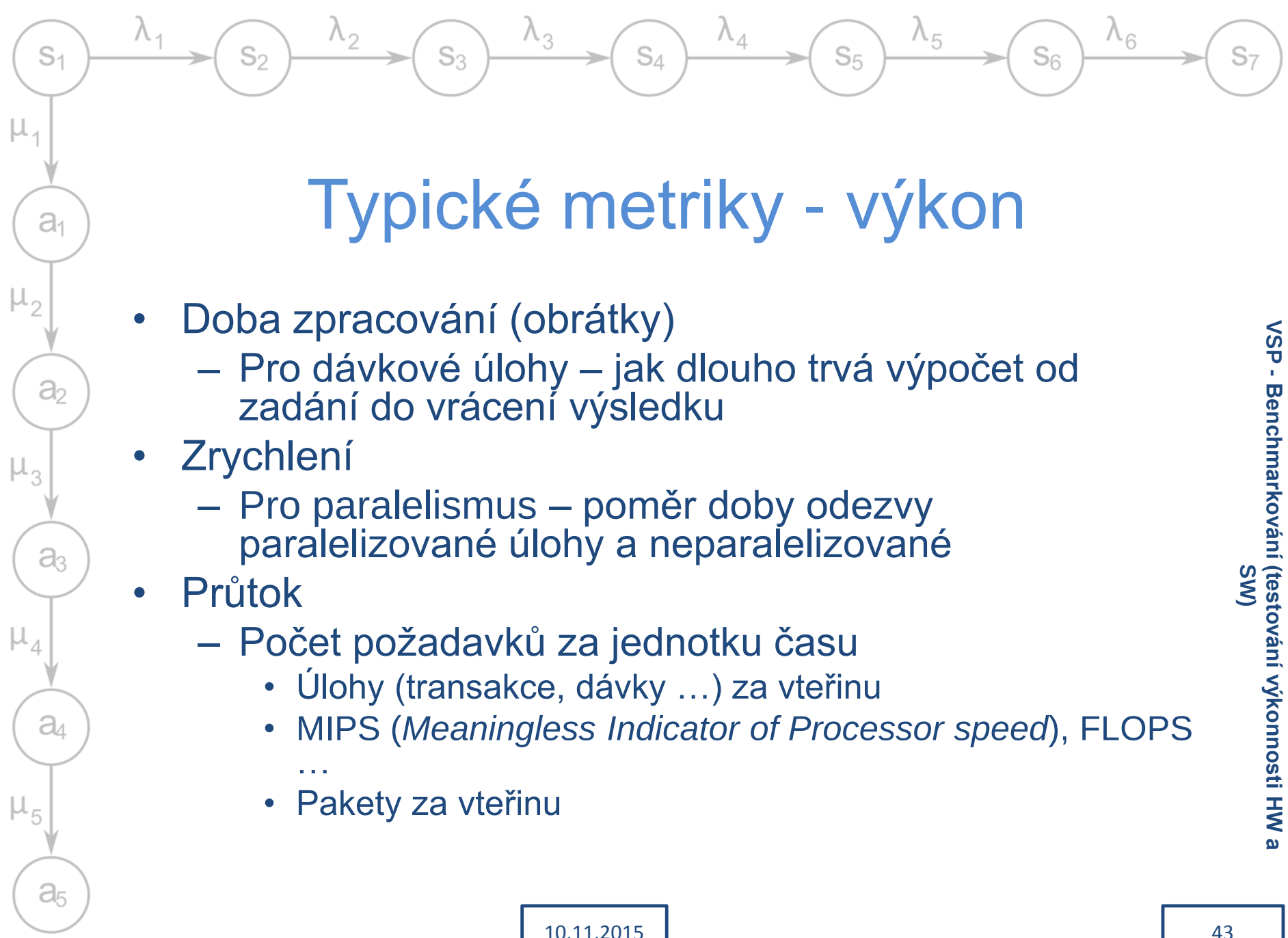


Typické metriky - kapacita

- Nominální (maximální) kapacita
  - Maximum dosažitelné v ideálních podmínkách
- Využitelná kapacita
  - Maximum dosažitelné se zachováním požadované doby odezvy
- Zlomová kapacita
  - Zátěž při které doba zůstává velmi nízká

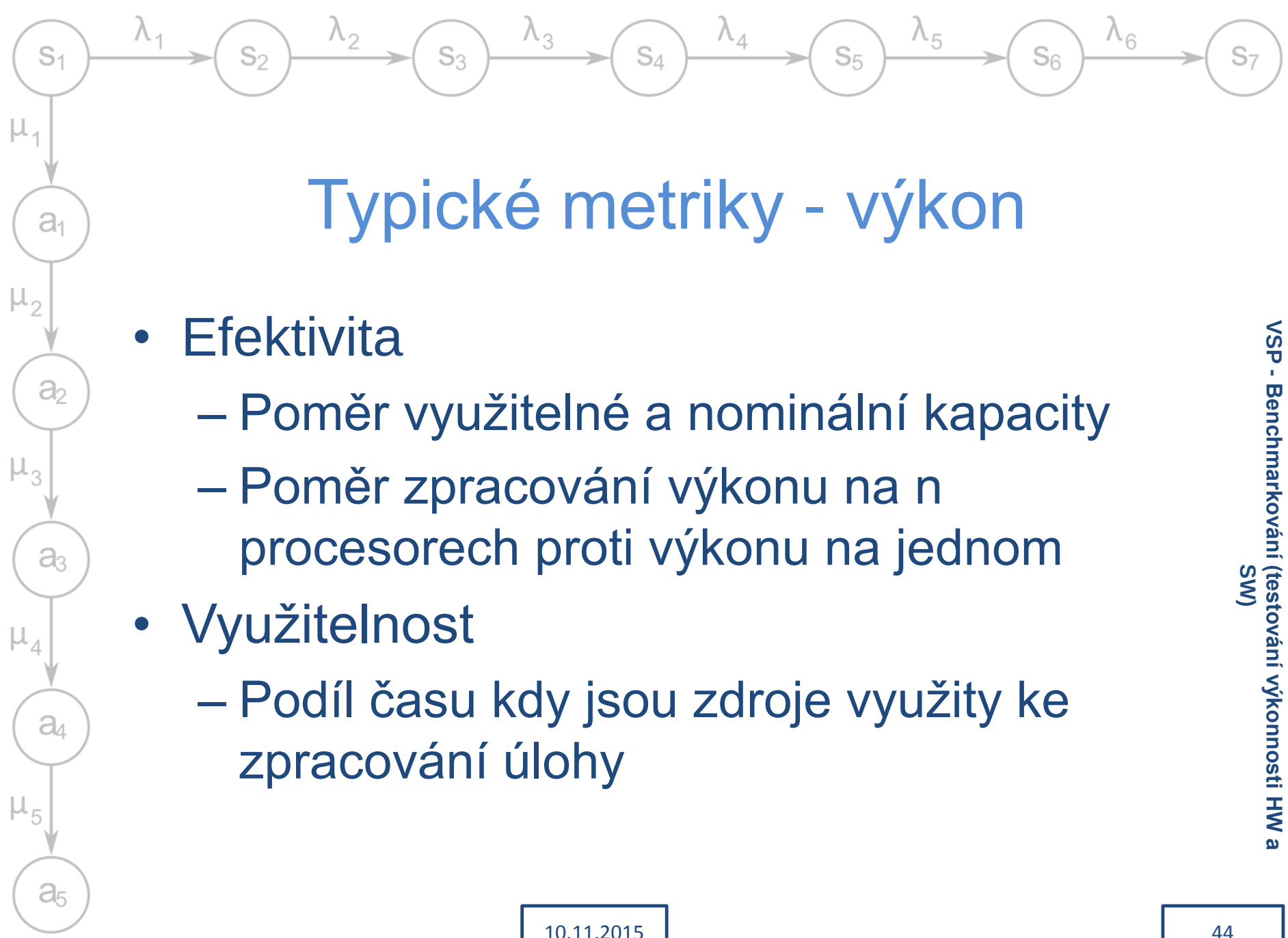


Typické metriky - kapacita



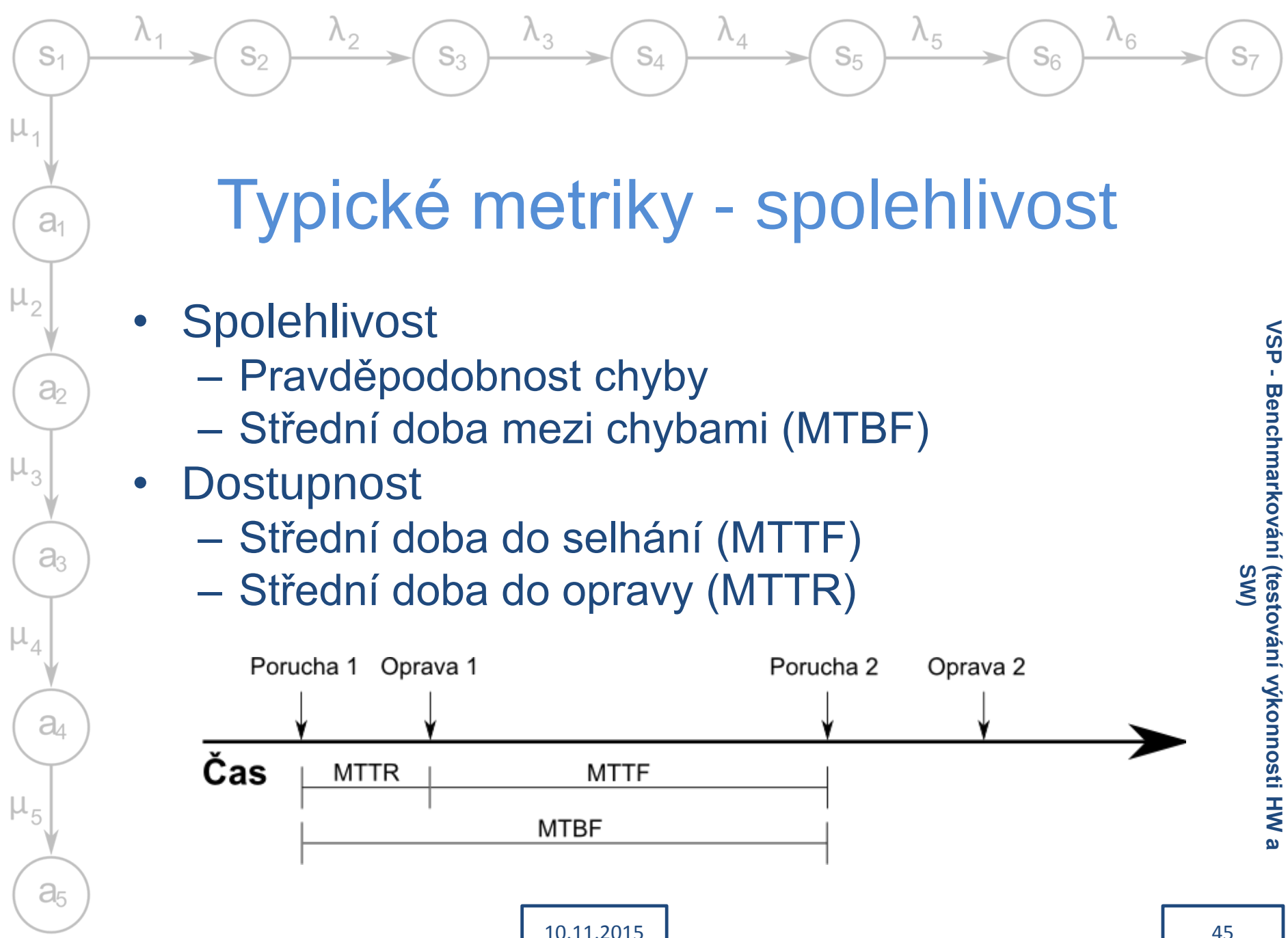
Typické metriky - výkon

- Doba zpracování (obrátky)
  - Pro dávkové úlohy – jak dlouho trvá výpočet od zadání do vrácení výsledku
- Zrychlení
  - Pro paralelismus – poměr doby odezvy paralelizované úlohy a neparalelizované
- Průtok
  - Počet požadavků za jednotku času
    - Úlohy (transakce, dávky …) za vteřinu
    - MIPS (*Meaningless Indicator of Processor speed*), FLOPS …
    - Pakety za vteřinu



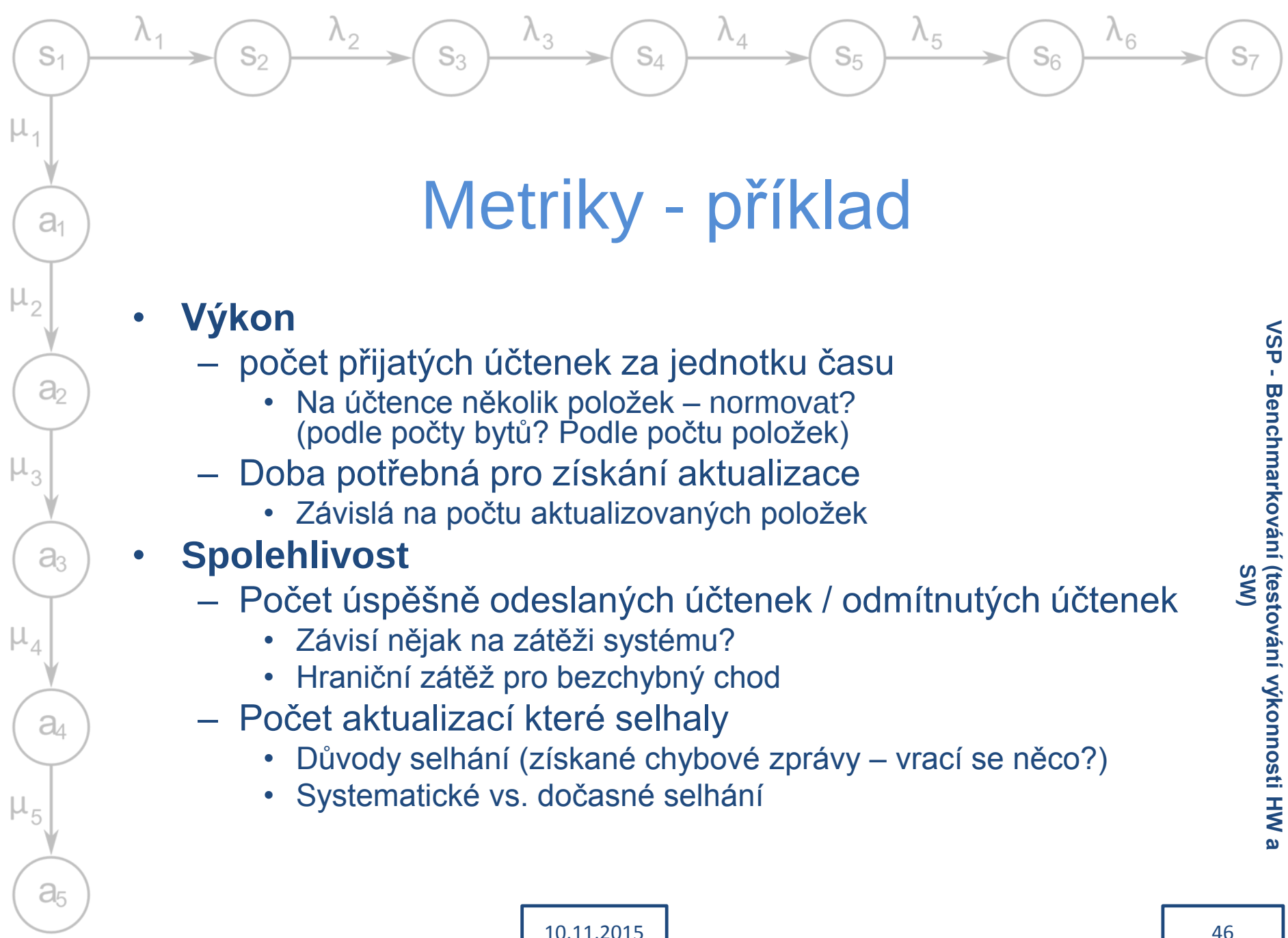
Typické metriky - výkon

- Efektivita
  - Poměr využitelné a nominální kapacity
  - Poměr zpracování výkonu na n procesorech proti výkonu na jednom
- Využitelnost
  - Podíl času kdy jsou zdroje využity ke zpracování úlohy



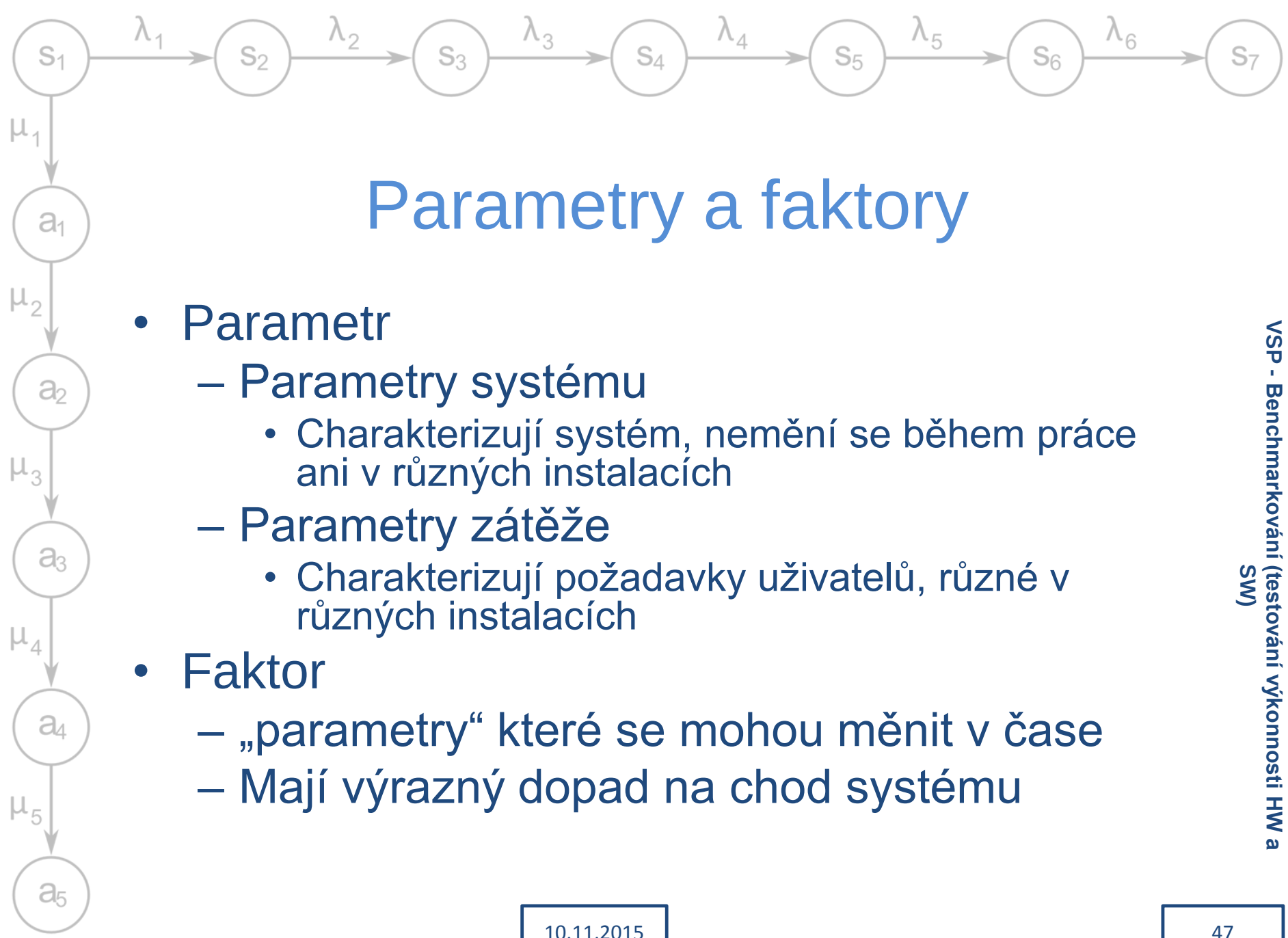
Typické metriky - spolehlivost

- Spolehlivost
  - Pravděpodobnost chyby
  - Střední doba mezi chybami (MTBF)
- Dostupnost
  - Střední doba do selhání (MTTF)
  - Střední doba do opravy (MTTR)



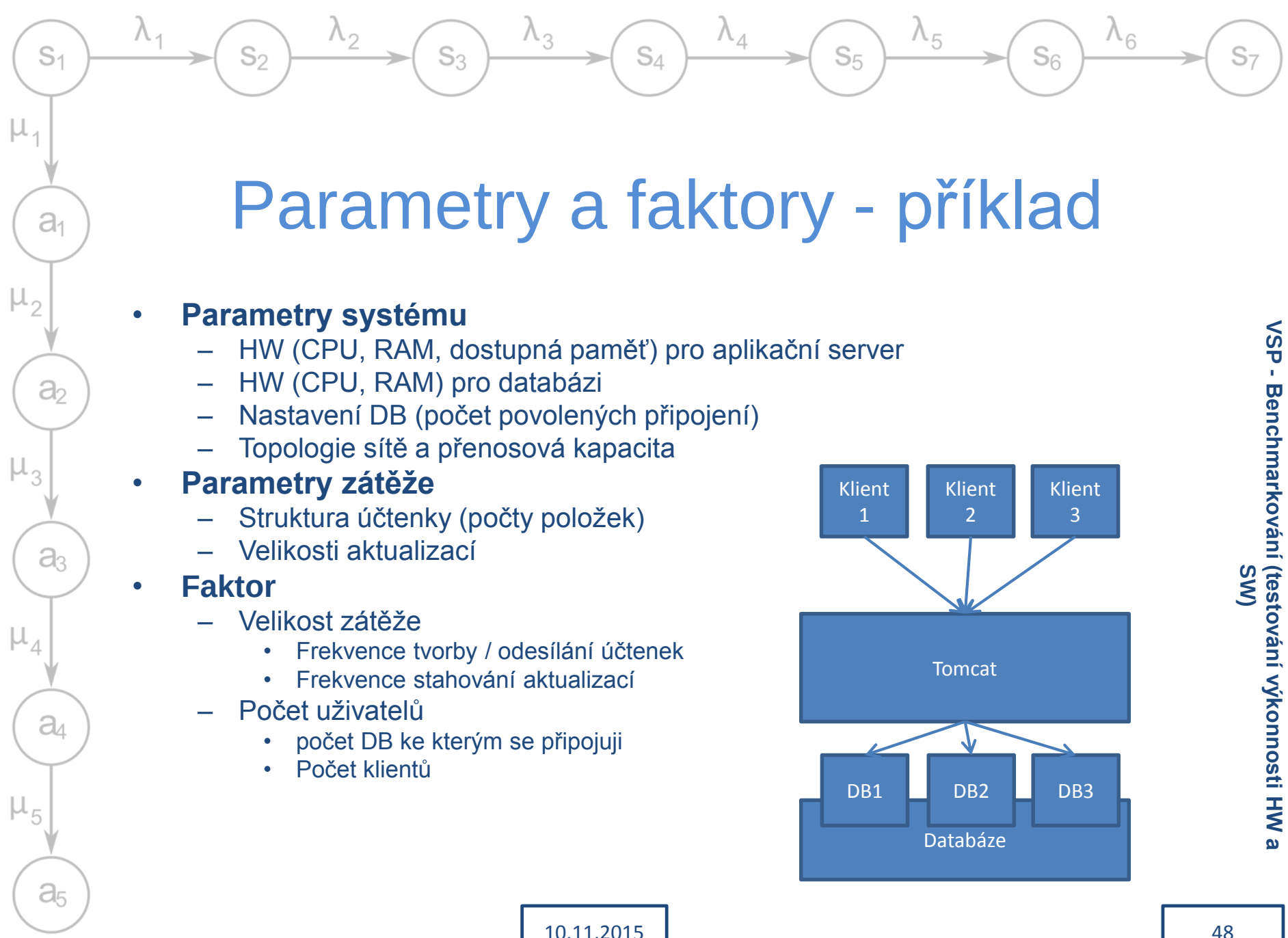
Metriky - příklad

- **Výkon**
  - počet přijatých účtenek za jednotku času
    - Na účtence několik položek – normovat? (podle počty bytů? Podle počtu položek)
  - Doba potřebná pro získání aktualizace
    - Závislá na počtu aktualizovaných položek
- **Spolehlivost**
  - Počet úspěšně odeslaných účtenek / odmítnutých účtenek
    - Závisí nějak na zátěži systému?
    - Hraniční zátěž pro bezchybný chod
  - Počet aktualizací které selhaly
    - Důvody selhání (získané chybové zprávy – vrací se něco?)
    - Systematické vs. dočasné selhání



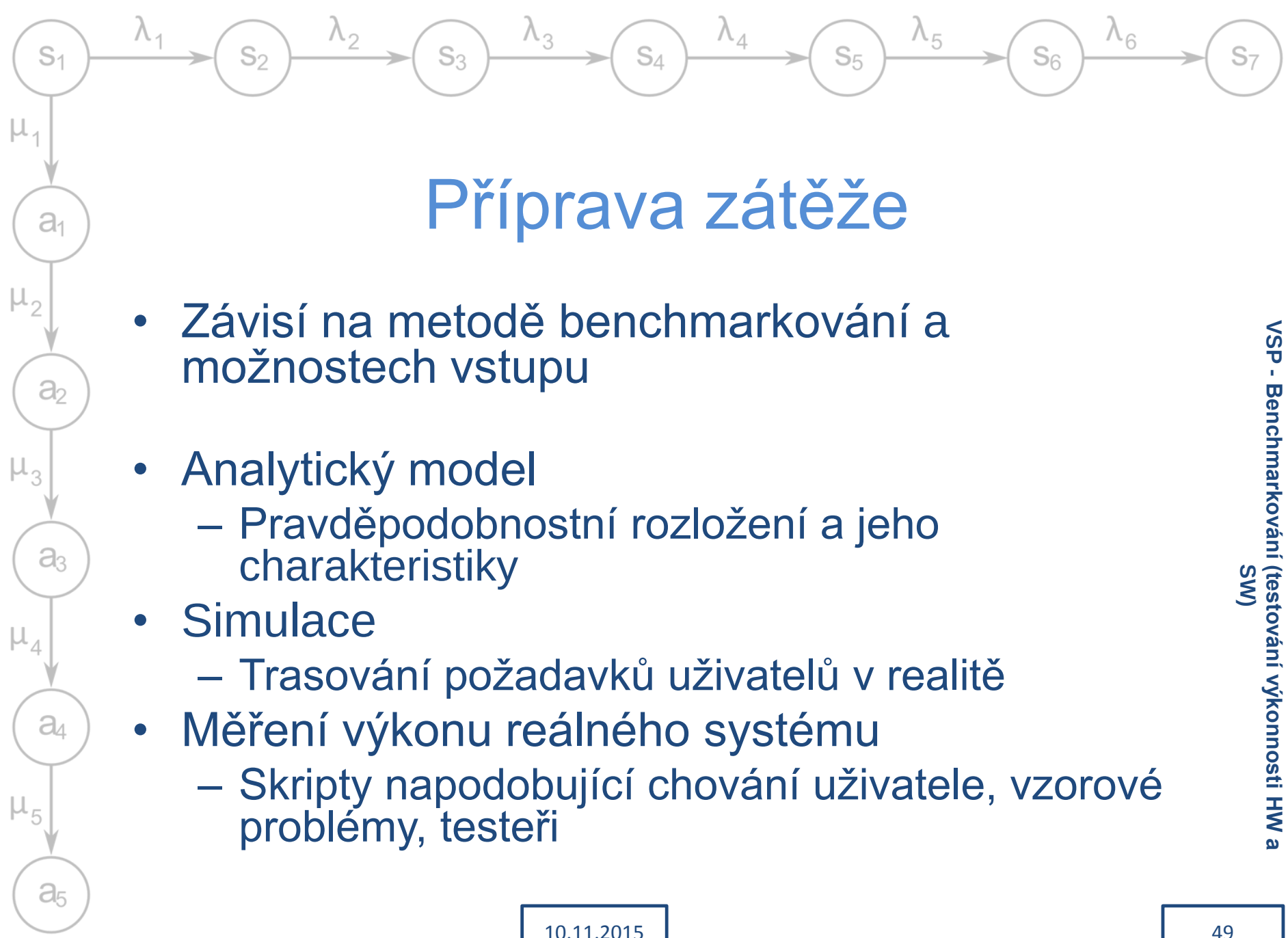
Parametry a faktory

- Parametr
  - Parametry systému
    - Charakterizují systém, nemění se během práce ani v různých instalacích
  - Parametry zátěže
    - Charakterizují požadavky uživatelů, různé v různých instalacích
- Faktor
  - „parametry“ které se mohou měnit v čase
  - Mají výrazný dopad na chod systému



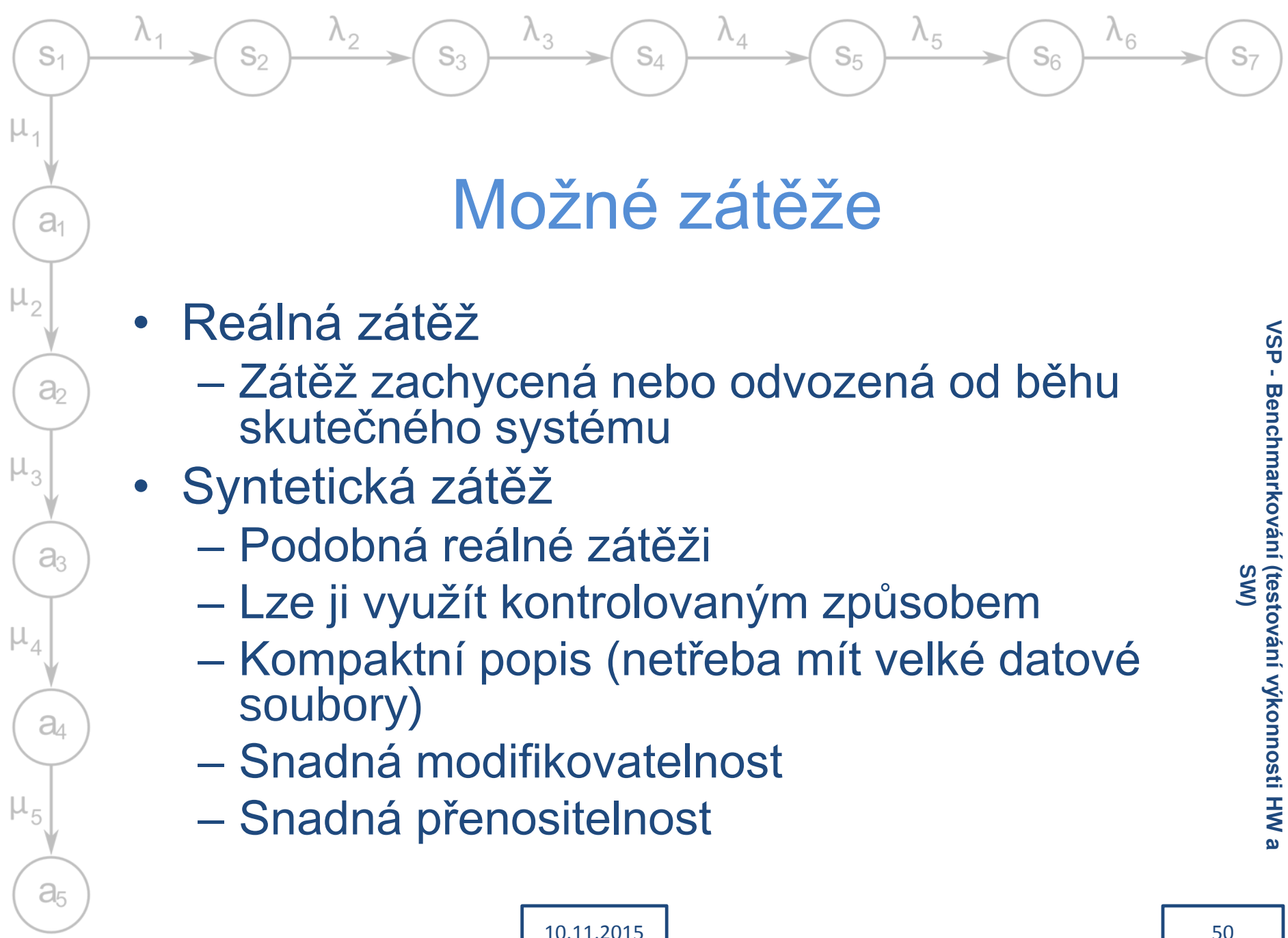
Parametry a faktory - příklad

- **Parametry systému**
  - HW (CPU, RAM, dostupná paměť) pro aplikační server
  - HW (CPU, RAM) pro databázi
  - Nastavení DB (počet povolených připojení)
  - Topologie sítě a přenosová kapacita
- **Parametry zátěže**
  - Struktura účtenky (počty položek)
  - Velikosti aktualizací
- **Faktor**
  - Velikost zátěže
    - Frekvence tvorby / odesílání účtenek
    - Frekvence stahování aktualizací
  - Počet uživatelů
    - počet DB ke kterým se připojuji
    - Počet klientů



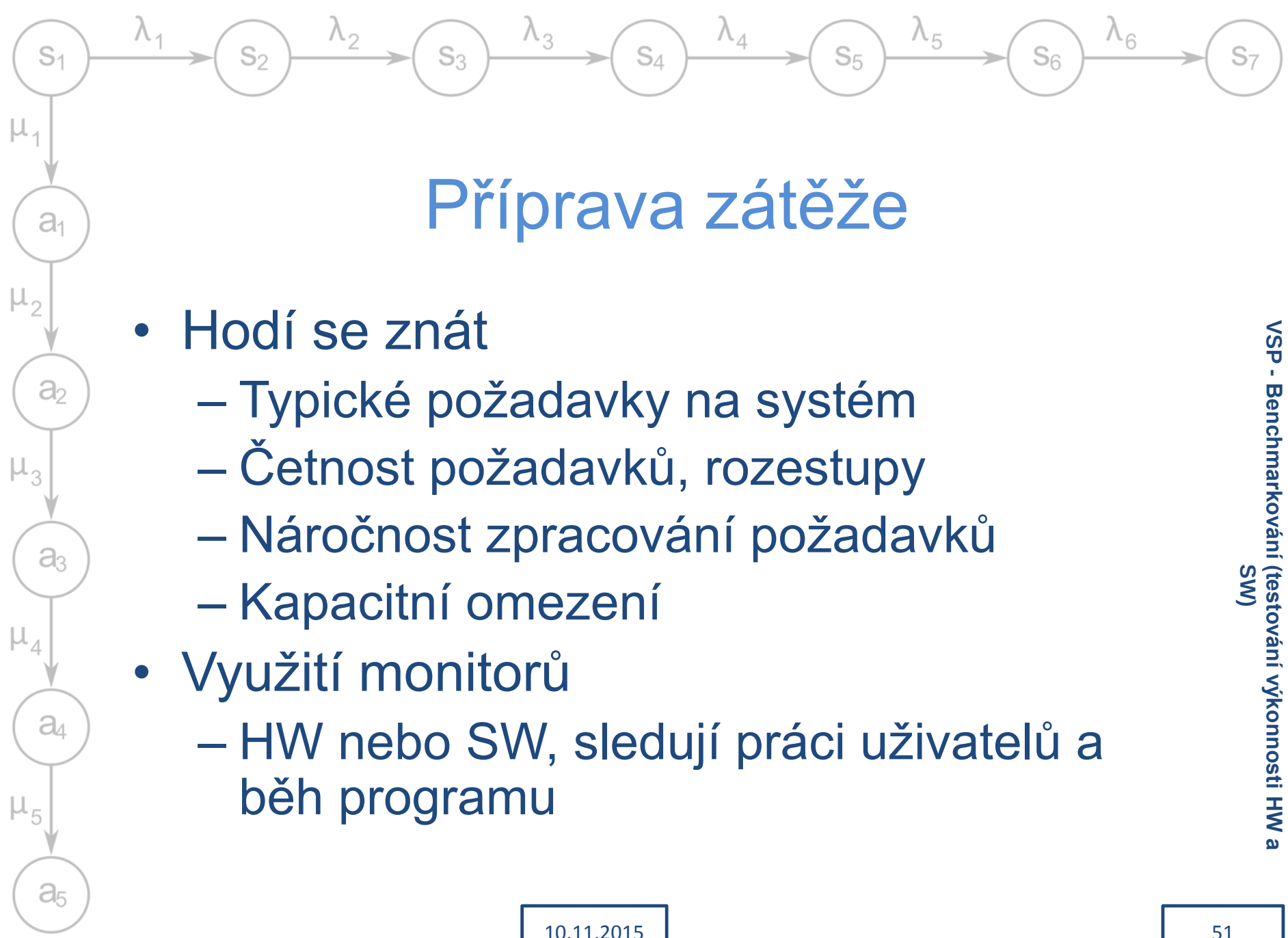
Příprava zátěže

- Závisí na metodě benchmarkování a možnostech vstupu
- Analytický model
  - Pravděpodobnostní rozložení a jeho charakteristiky
- Simulace
  - Trasování požadavků uživatelů v realitě
- Měření výkonu reálného systému
  - Skripty napodobující chování uživatele, vzorové problémy, testeři



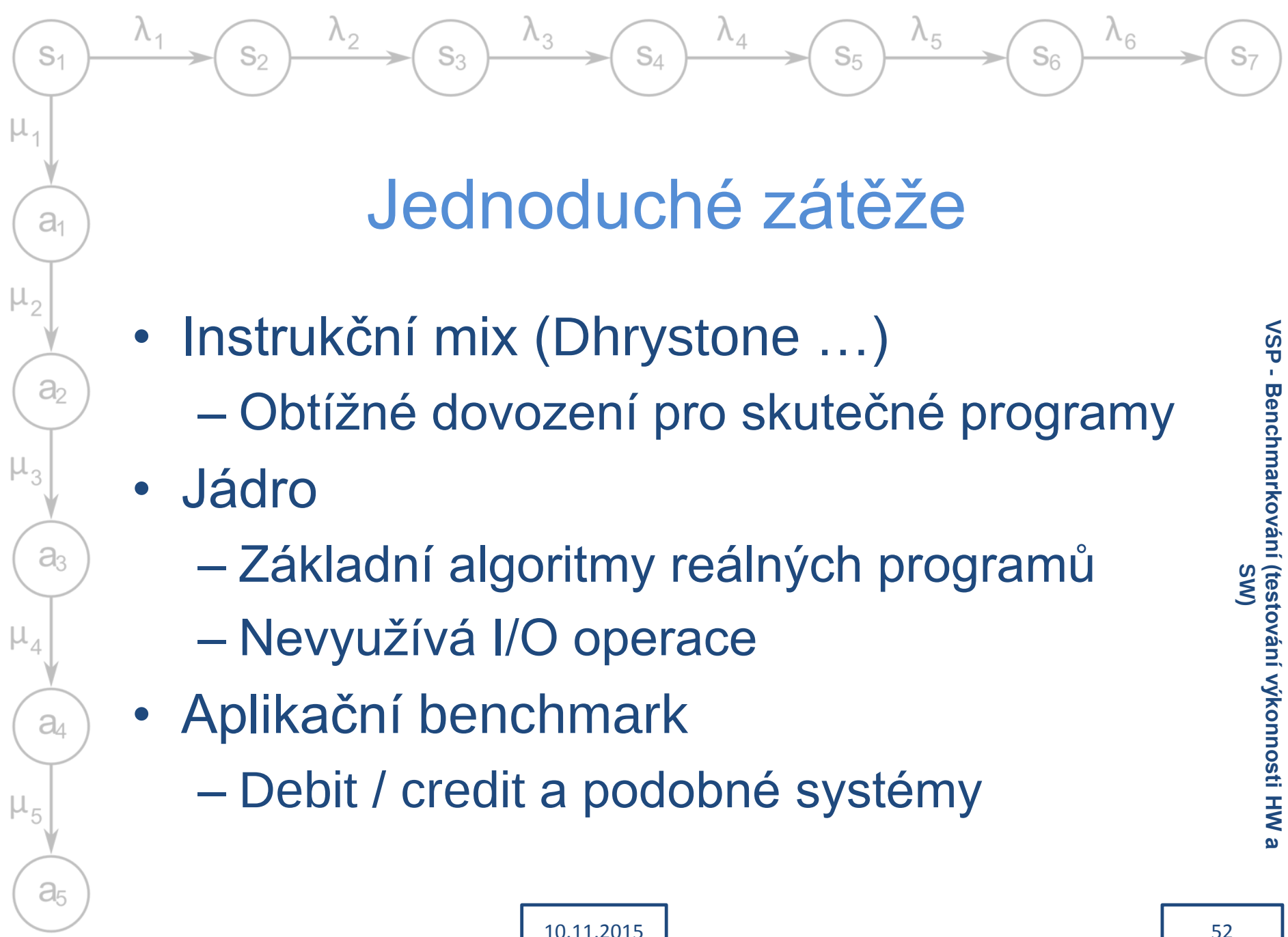
Možné zátěže

- Reálná zátěž
  - Zátěž zachycená nebo odvozená od běhu skutečného systému
- Syntetická zátěž
  - Podobná reálné zátěži
  - Lze ji využít kontrolovaným způsobem
  - Kompaktní popis (netřeba mít velké datové soubory)
  - Snadná modifikovatelnost
  - Snadná přenositelnost



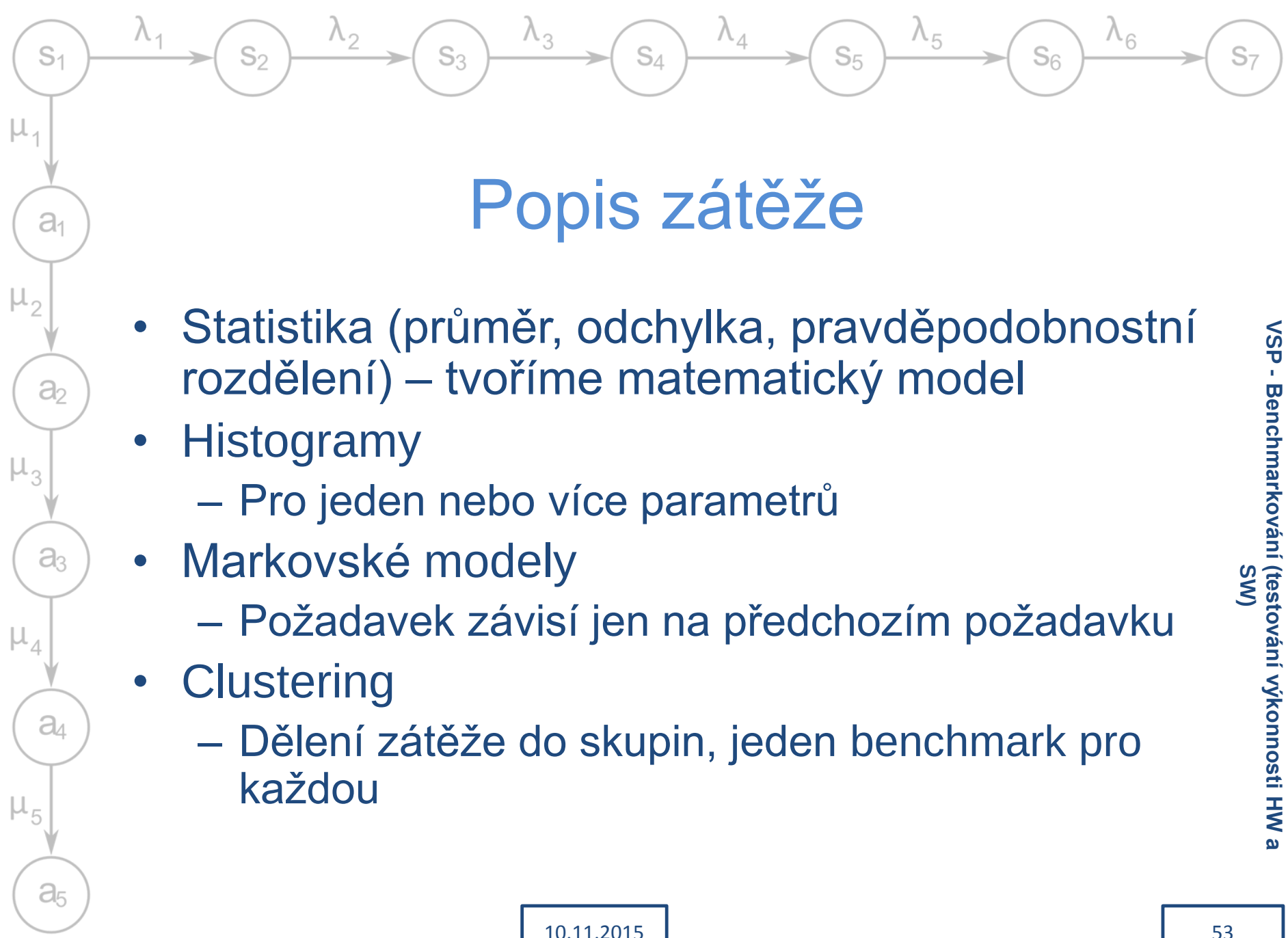
Příprava zátěže

- Hodí se znát
  - Typické požadavky na systém
  - Četnost požadavků, rozestupy
  - Náročnost zpracování požadavků
  - Kapacitní omezení
- Využití monitorů
  - HW nebo SW, sledují práci uživatelů a běh programu



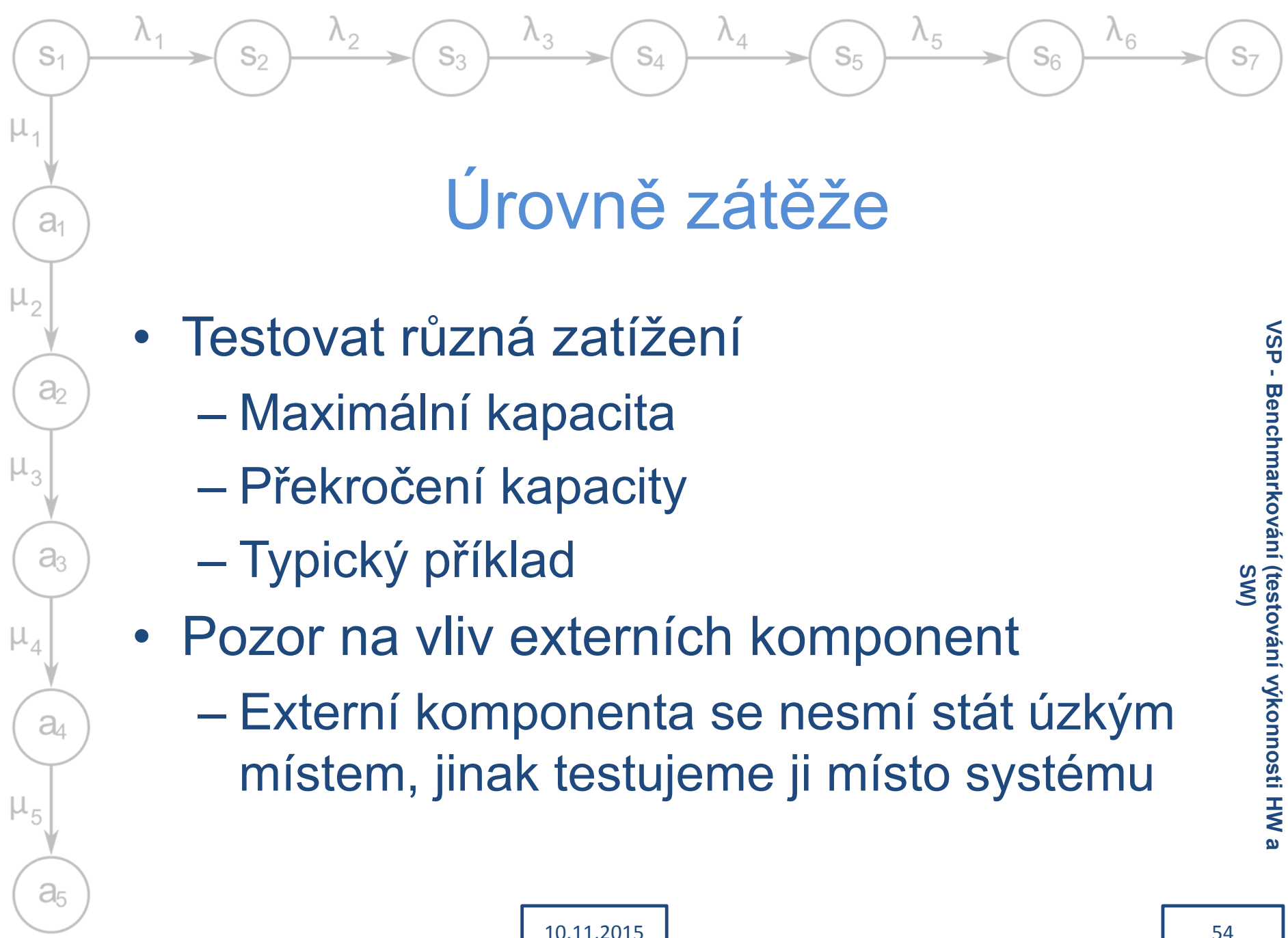
Jednoduché zátěže

- Instrukční mix (Dhrystone …)
  - Obtížné dovození pro skutečné programy
- Jádro
  - Základní algoritmy reálných programů
  - Nevyužívá I/O operace
- Aplikační benchmark
  - Debit / credit a podobné systémy



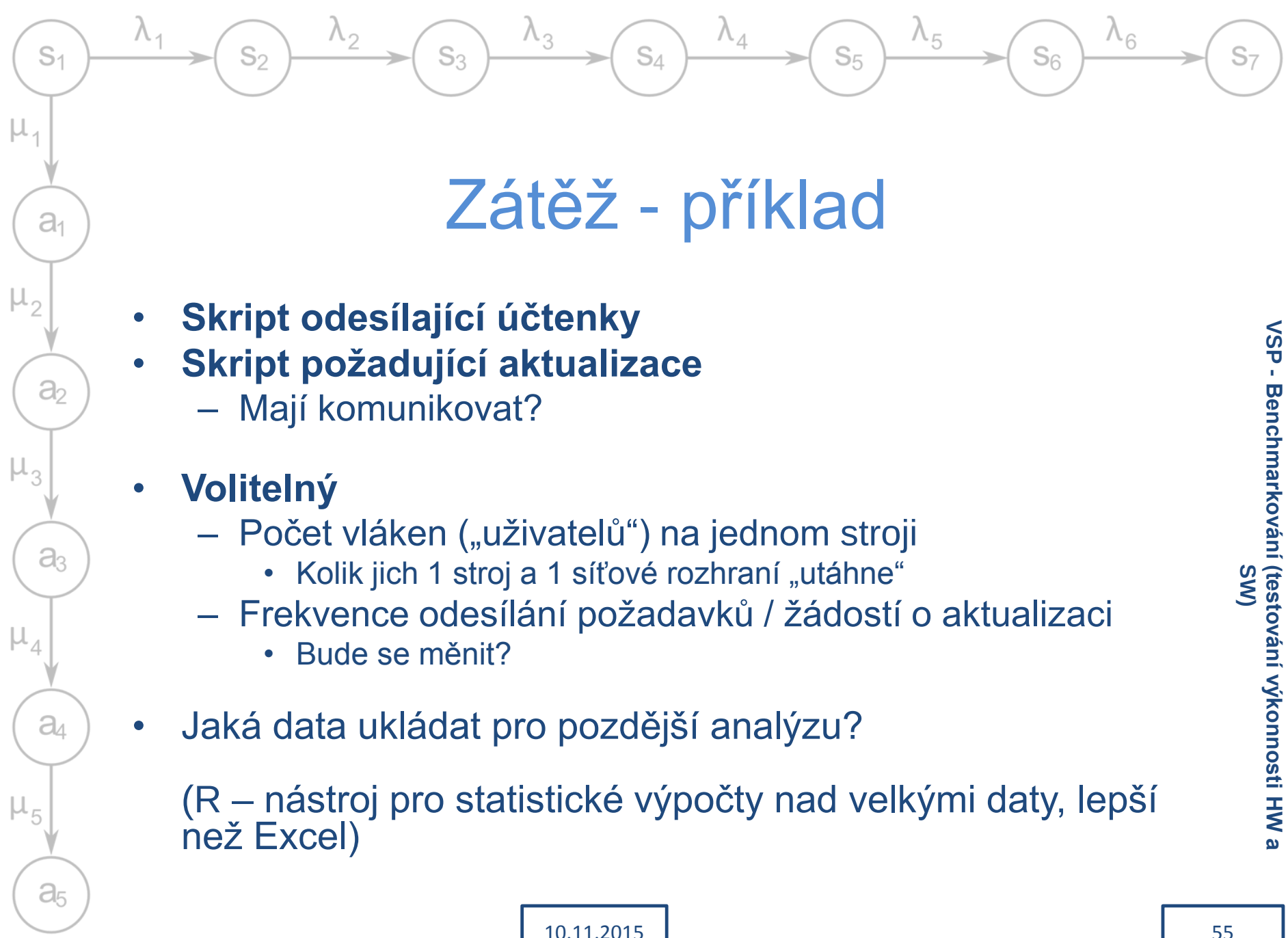
Popis zátěže

- Statistika (průměr, odchylka, pravděpodobnostní rozdělení) – tvoříme matematický model
- Histogramy
  - Pro jeden nebo více parametrů
- Markovské modely
  - Požadavek závisí jen na předchozím požadavku
- Clustering
  - Dělení zátěže do skupin, jeden benchmark pro každou



Úrovně zátěže

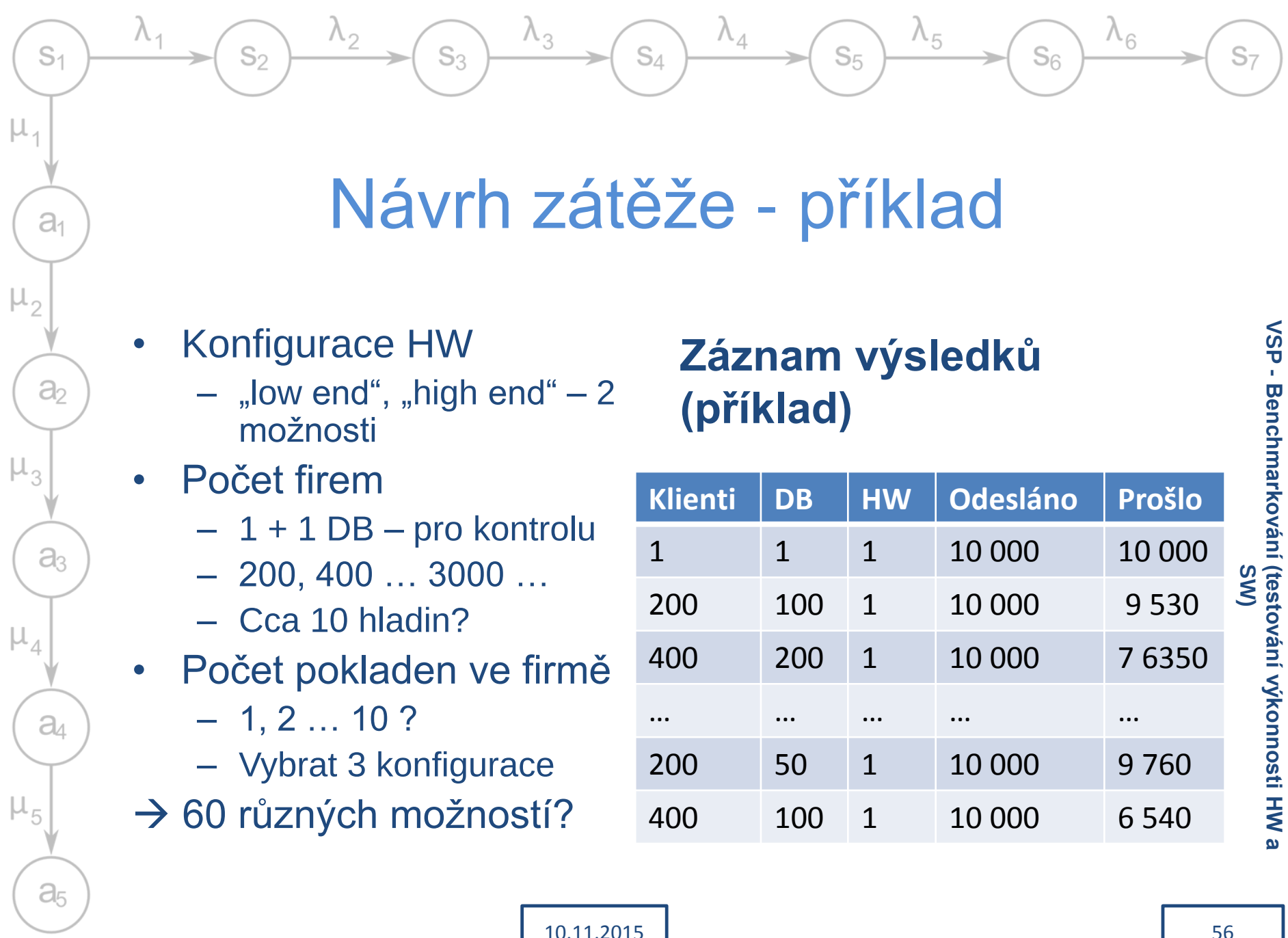
- Testovat různá zatížení
  - Maximální kapacita
  - Překročení kapacity
  - Typický příklad
- Pozor na vliv externích komponent
  - Externí komponenta se nesmí stát úzkým místem, jinak testujeme ji místo systému



Zátěž - příklad

- **Skript odesílající účtenky**
- **Skript požadující aktualizace**
  - Mají komunikovat?
- **Volitelný**
  - Počet vláken („uživatelů“) na jednom stroji
    - Kolik jich 1 stroj a 1 síťové rozhraní „utáhne“
  - Frekvence odesílání požadavků / žádostí o aktualizaci
    - Bude se měnit?
- Jaká data ukládat pro pozdější analýzu?

  (R – nástroj pro statistické výpočty nad velkými daty, lepší než Excel)



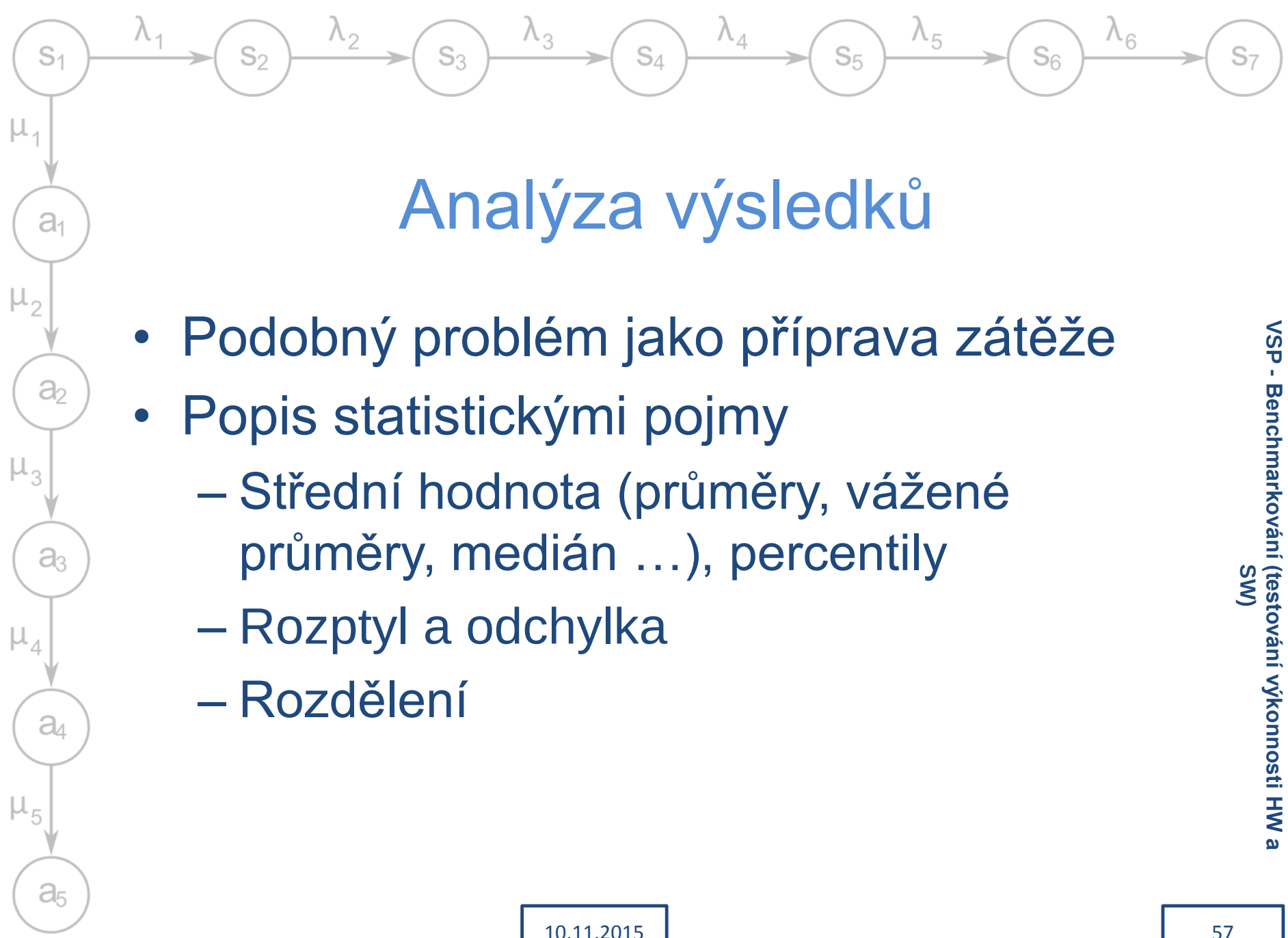
Návrh zátěže - příklad

- Konfigurace HW
  - „low end“, „high end“ – 2 možnosti
- Počet firem
  - 1 + 1 DB – pro kontrolu
  - 200, 400 … 3000 …
  - Cca 10 hladin?
- Počet pokladen ve firmě
  - 1, 2 … 10 ?
  - Vybrat 3 konfigurace

→ 60 různých možností?

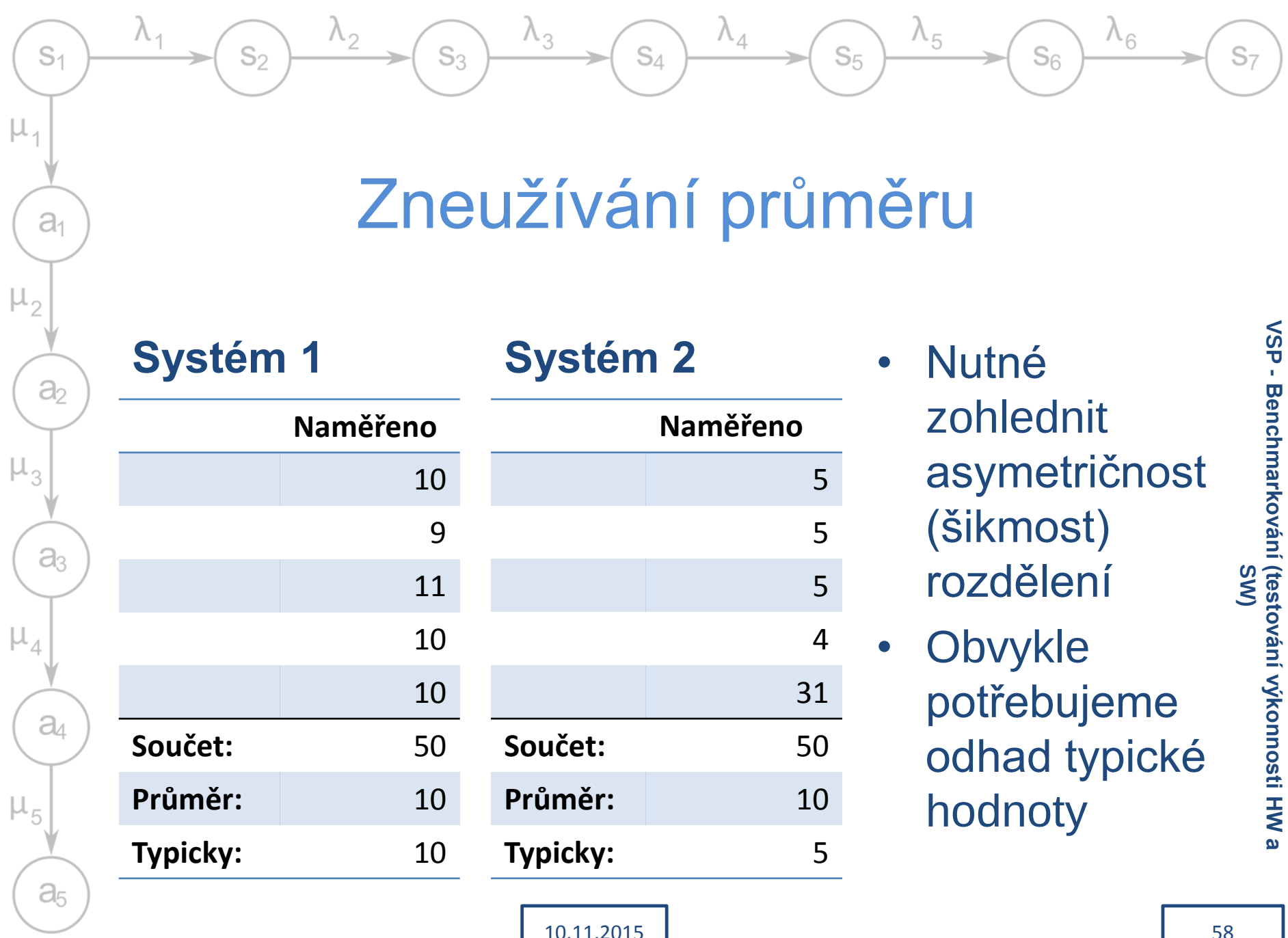
Záznam výsledků (příklad)

Klienti	DB	HW	Odesláno	Prošlo
1	1	1	10 000	10 000
200	100	1	10 000	9 530
400	200	1	10 000	7 6350
…	…	…	…	…
200	50	1	10 000	9 760
400	100	1	10 000	6 540



Analýza výsledků

- Podobný problém jako příprava zátěže
- Popis statistickými pojmy
  - Střední hodnota (průměry, vážené průměry, medián …), percentily
  - Rozptyl a odchylka
  - Rozdělení



Zneužívání průměru

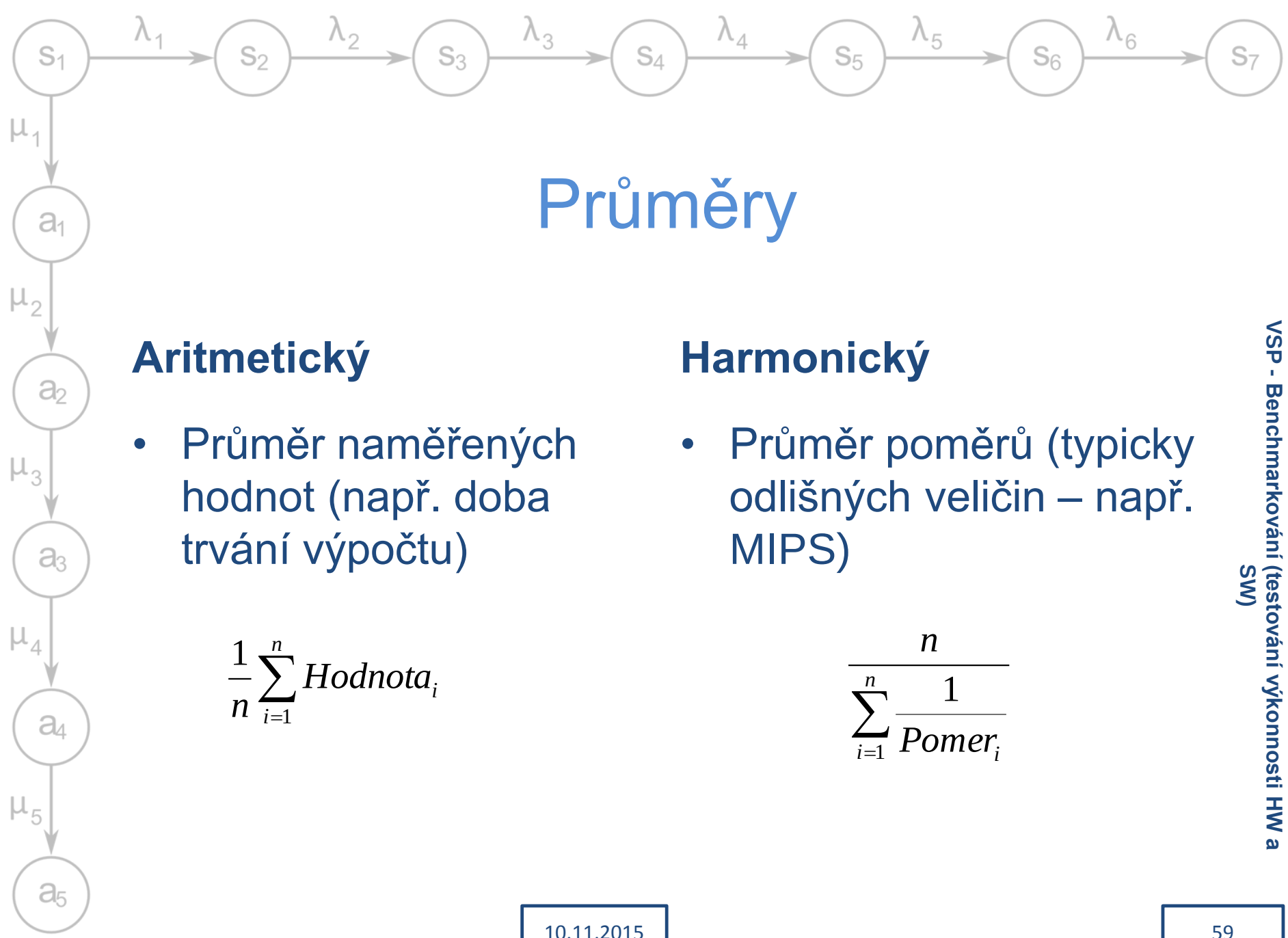
Systém 1

	Naměřeno
	10
	9
	11
	10
	10
Součet:	50
Průměr:	10
Typicky:	10

Systém 2

	Naměřeno
	5
	5
	5
	4
	31
Součet:	50
Průměr:	10
Typicky:	5

- Nutné zohlednit asymetričnost (šikmost) rozdělení
- Obvykle potřebujeme odhad typické hodnoty



Průměry

Aritmetický

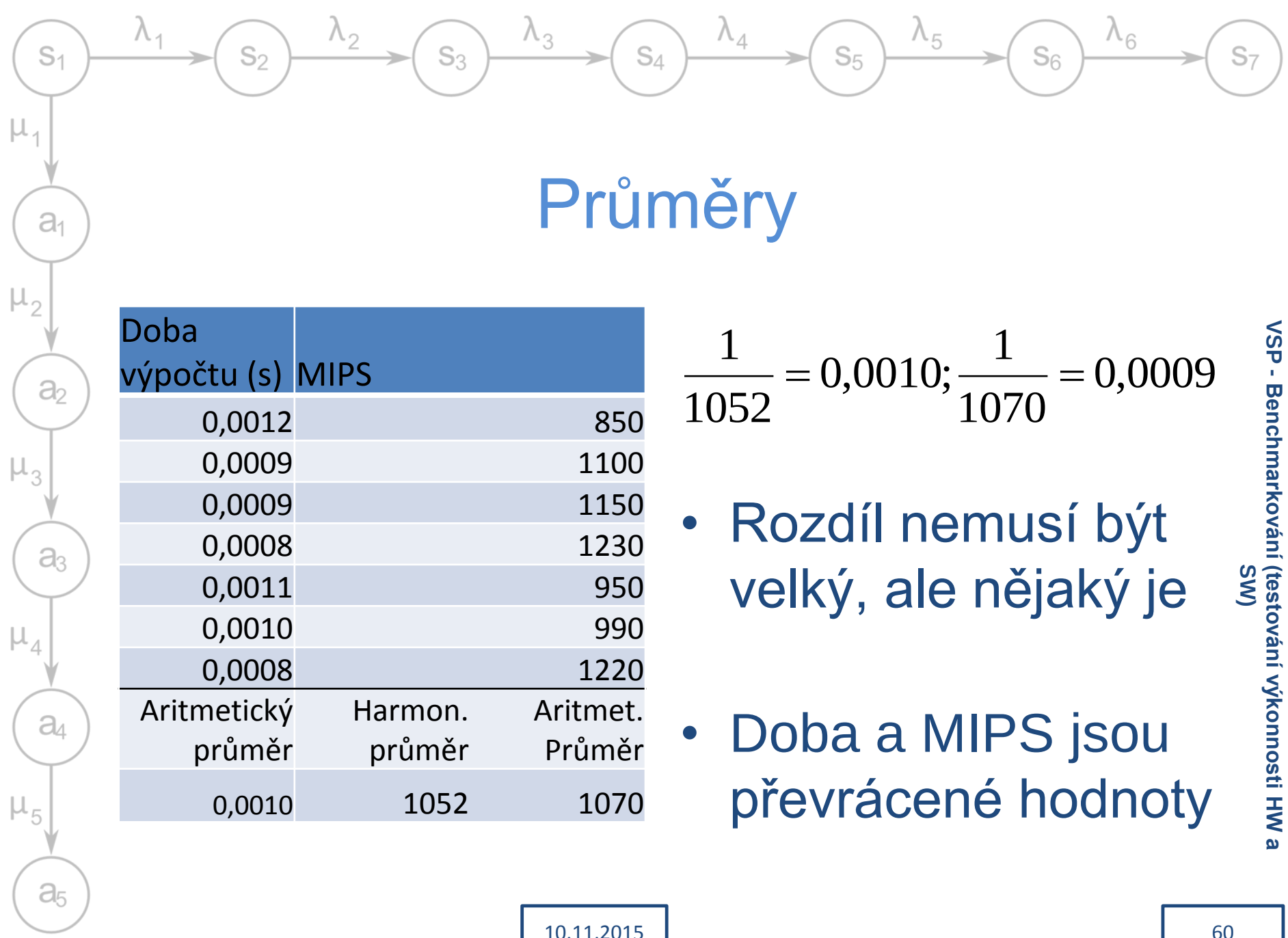
- Průměr naměřených hodnot (např. doba trvání výpočtu)

$$\frac{1}{n}\sum_{i=1}^{n} Hodnota_i$$

Harmonický

- Průměr poměrů (typicky odlišných veličin – např. MIPS)

$$\frac{n}{\sum_{i=1}^{n} \frac{1}{Pomer_i}}$$

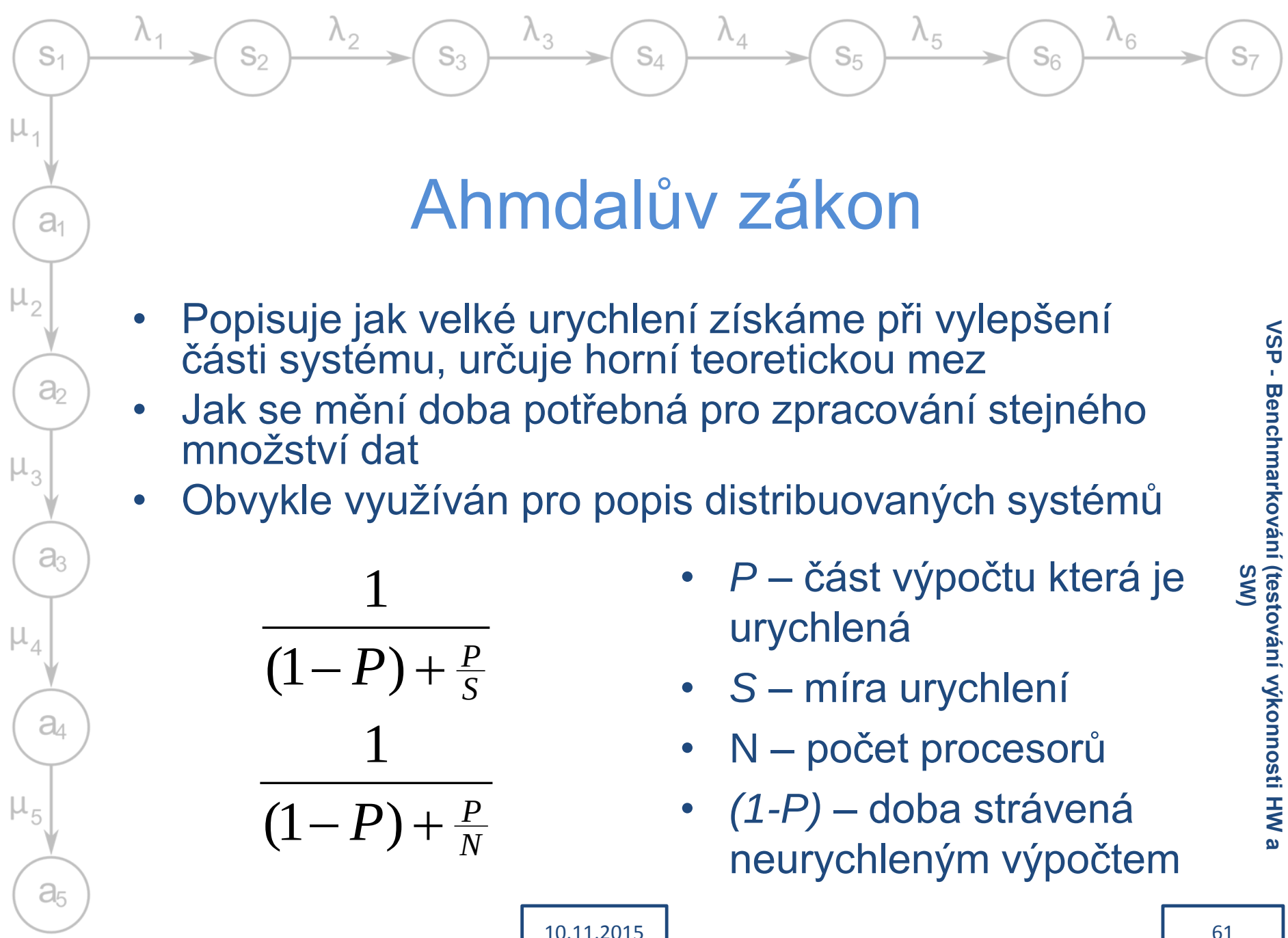


Průměry

Doba výpočtu (s)	MIPS	
0,0012		850
0,0009		1100
0,0009		1150
0,0008		1230
0,0011		950
0,0010		990
0,0008		1220
Aritmetický průměr	Harmon. průměr	Aritmet. Průměr
0,0010	1052	1070

$$\frac{1}{1052} = 0{,}0010; \frac{1}{1070} = 0{,}0009$$

- Rozdíl nemusí být velký, ale nějaký je
- Doba a MIPS jsou převrácené hodnoty



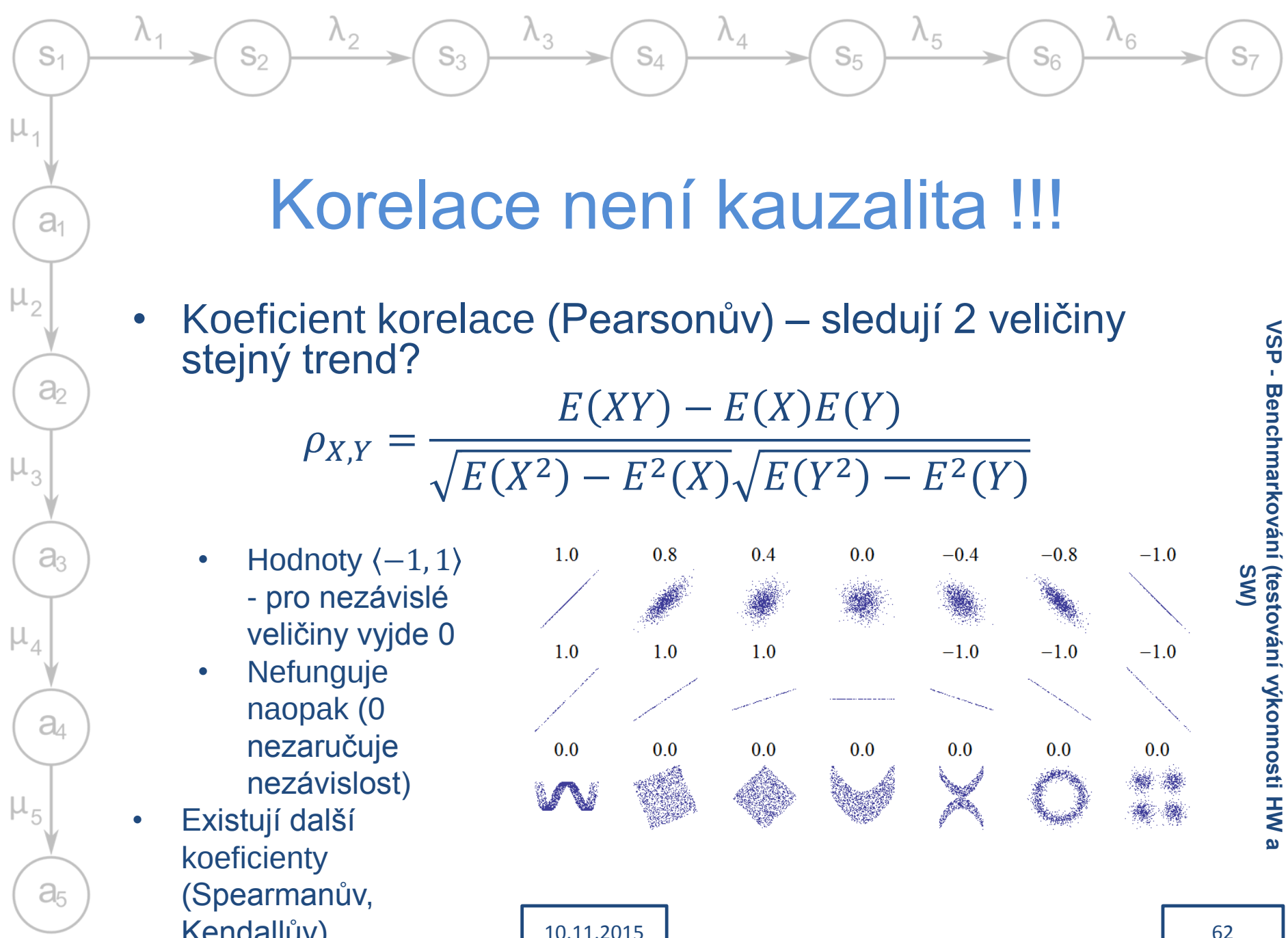
Ahmdalův zákon

- Popisuje jak velké urychlení získáme při vylepšení části systému, určuje horní teoretickou mez
- Jak se mění doba potřebná pro zpracování stejného množství dat
- Obvykle využíván pro popis distribuovaných systémů

$$\frac{1}{(1-P)+\frac{P}{S}}$$

$$\frac{1}{(1-P)+\frac{P}{N}}$$

- *P* – část výpočtu která je urychlená
- *S* – míra urychlení
- N – počet procesorů
- *(1-P)* – doba strávená neurychleným výpočtem



Korelace není kauzalita !!!

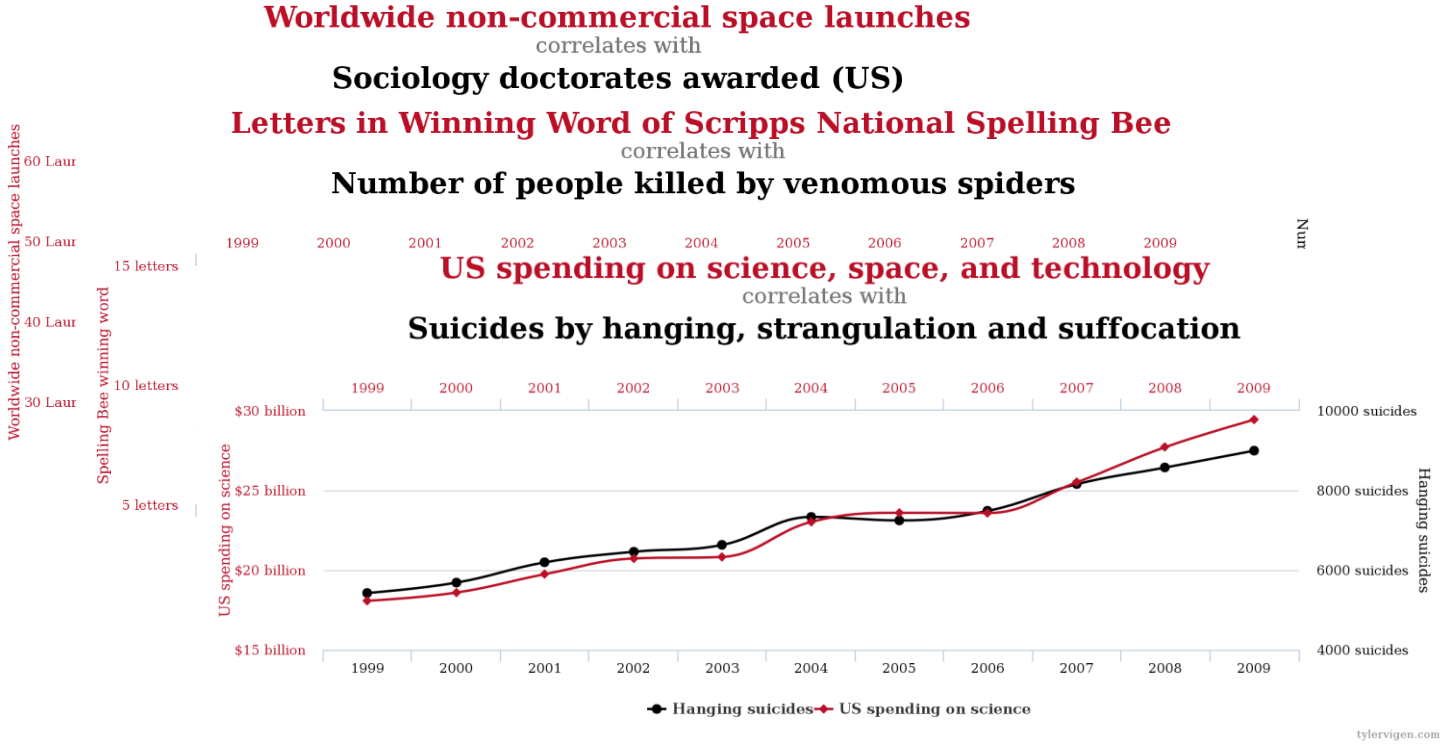
- Koeficient korelace (Pearsonův) – sledují 2 veličiny stejný trend?

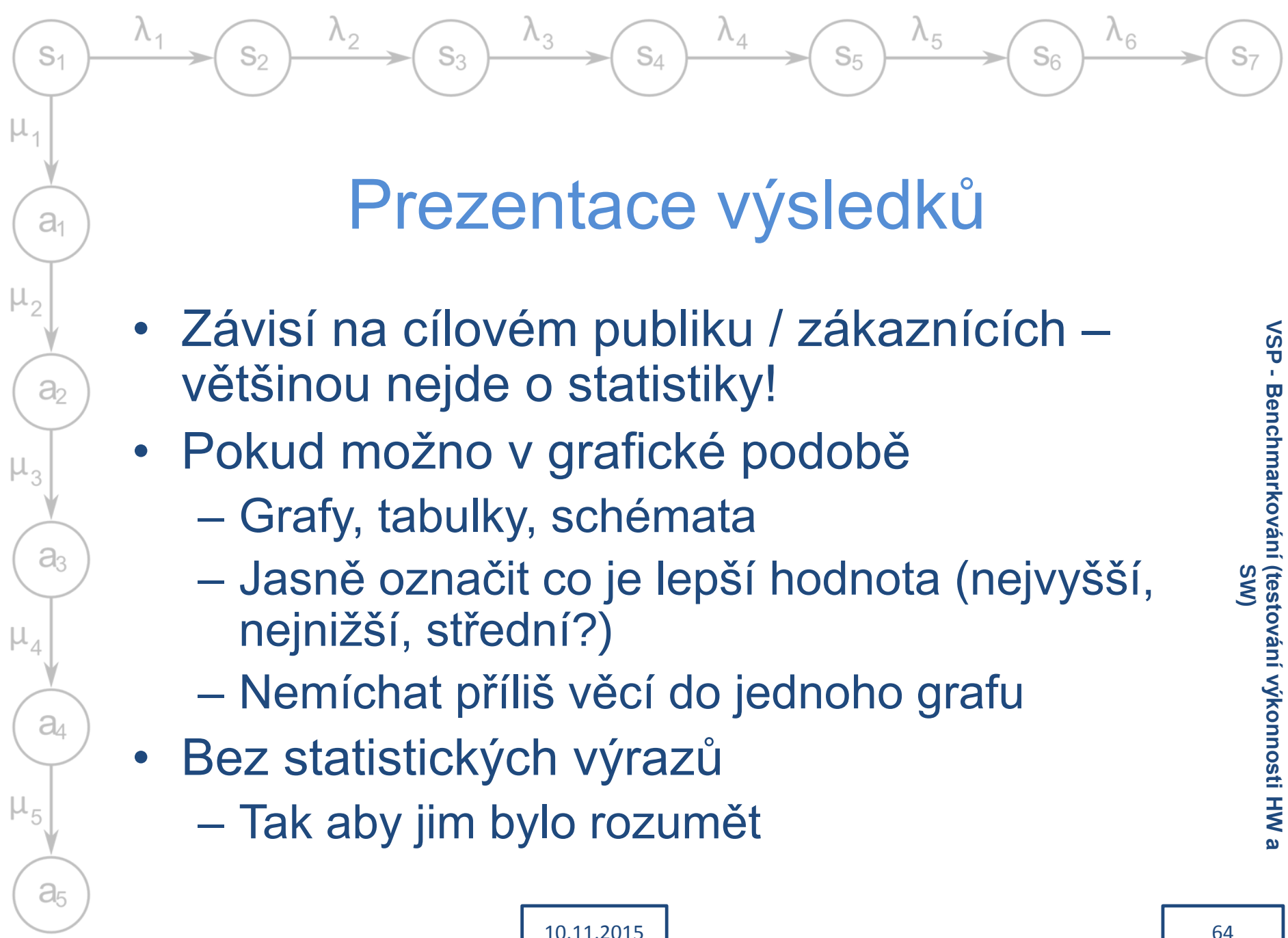
$$\rho_{X,Y} = \frac{E(XY) - E(X)E(Y)}{\sqrt{E(X^2) - E^2(X)}\sqrt{E(Y^2) - E^2(Y)}}$$

  - Hodnoty $\langle -1, 1 \rangle$ - pro nezávislé veličiny vyjde 0
  - Nefunguje naopak (0 nezaručuje nezávislost)
- Existují další koeficienty (Spearmanův, Kendallův)

Korelace není kauzalita

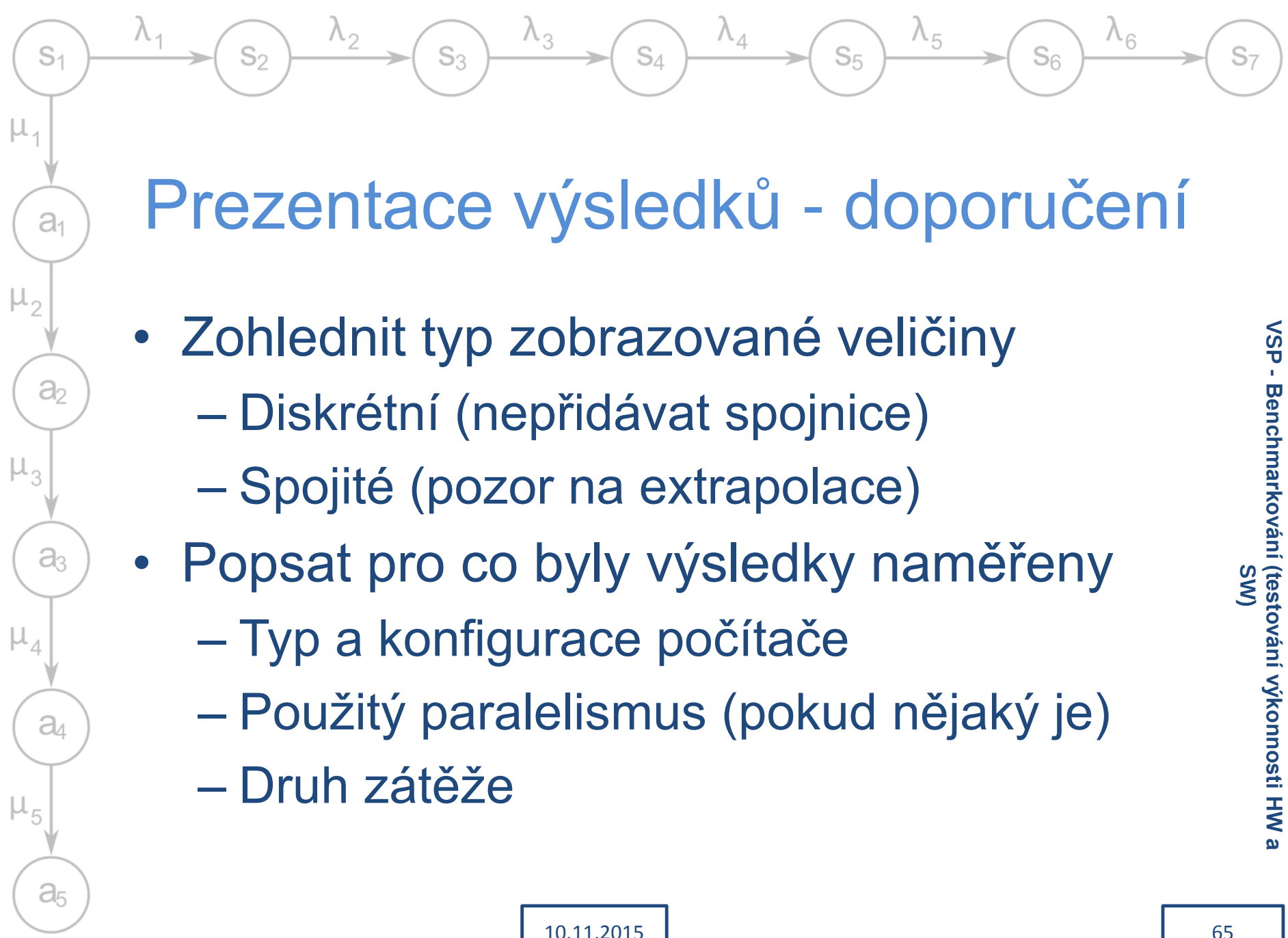
- Náhodné korelace - http://www.tylervigen.com/spurious-correlations





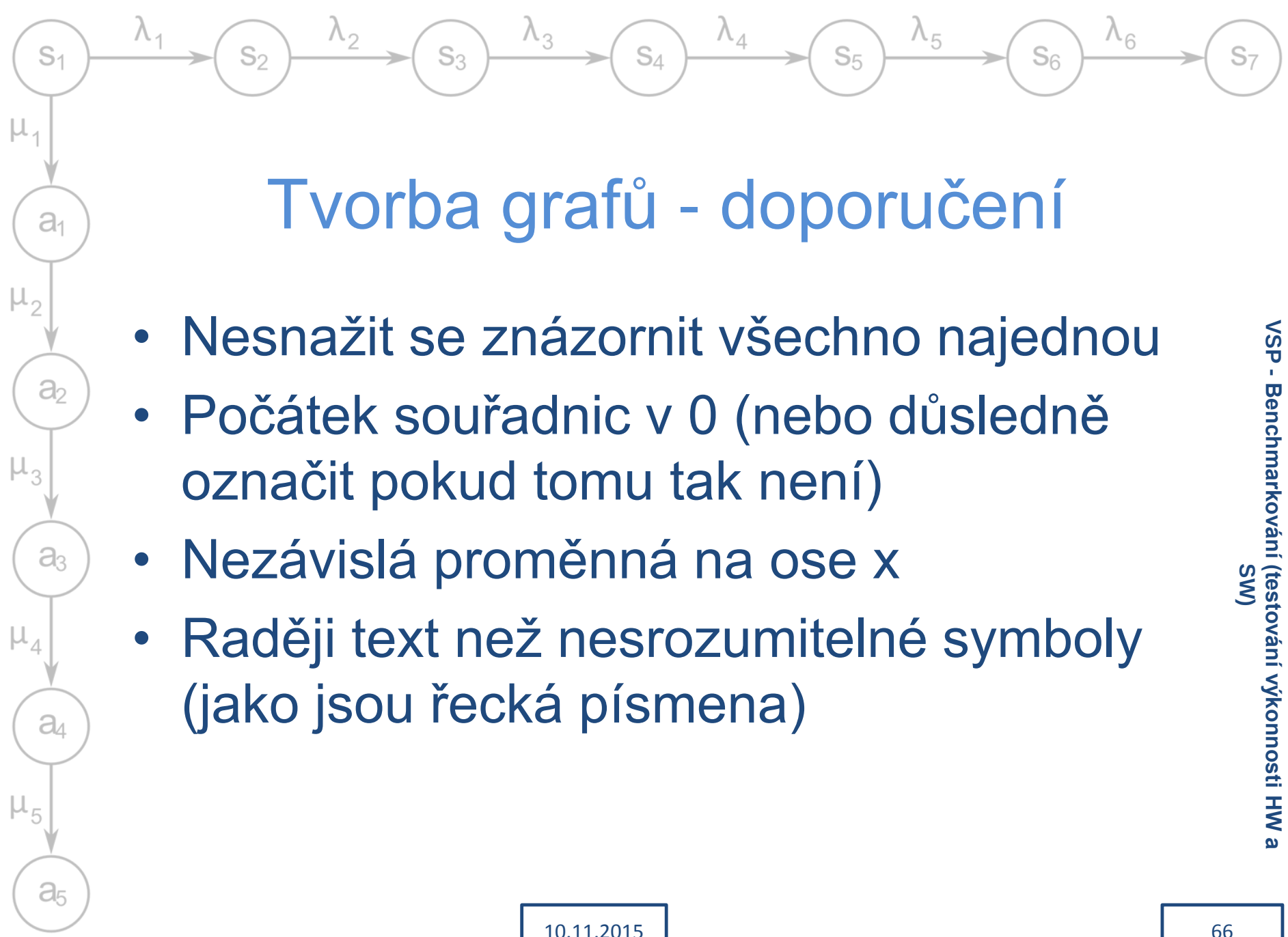
Prezentace výsledků

- Závisí na cílovém publiku / zákaznících – většinou nejde o statistiky!
- Pokud možno v grafické podobě
  - Grafy, tabulky, schémata
  - Jasně označit co je lepší hodnota (nejvyšší, nejnižší, střední?)
  - Nemíchat příliš věcí do jednoho grafu
- Bez statistických výrazů
  - Tak aby jim bylo rozumět



Prezentace výsledků - doporučení

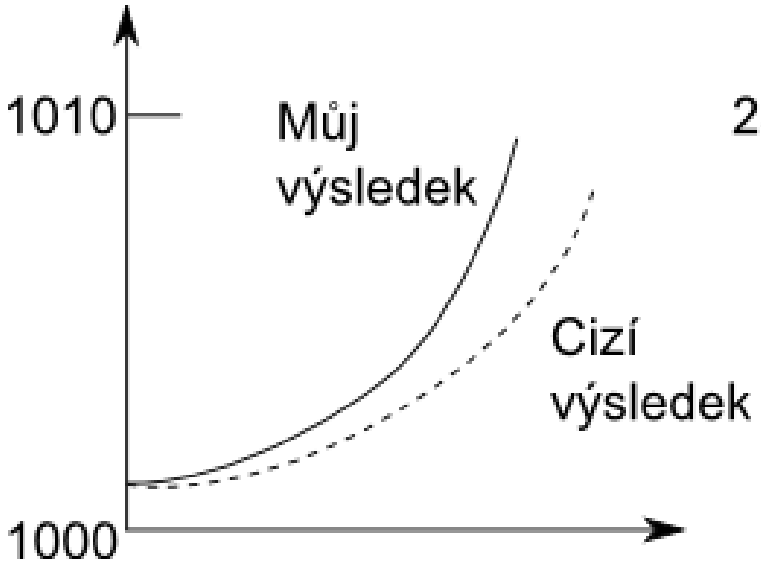
- Zohlednit typ zobrazované veličiny
  - Diskrétní (nepřidávat spojnice)
  - Spojité (pozor na extrapolace)
- Popsat pro co byly výsledky naměřeny
  - Typ a konfigurace počítače
  - Použitý paralelismus (pokud nějaký je)
  - Druh zátěže

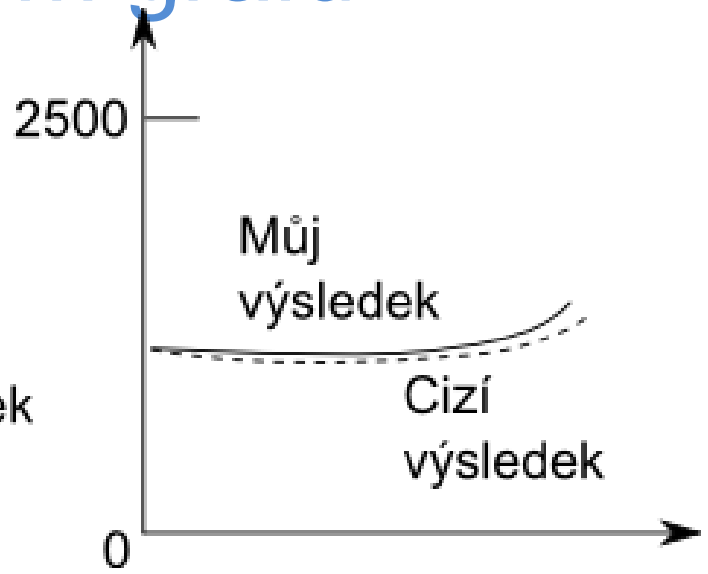


Tvorba grafů - doporučení

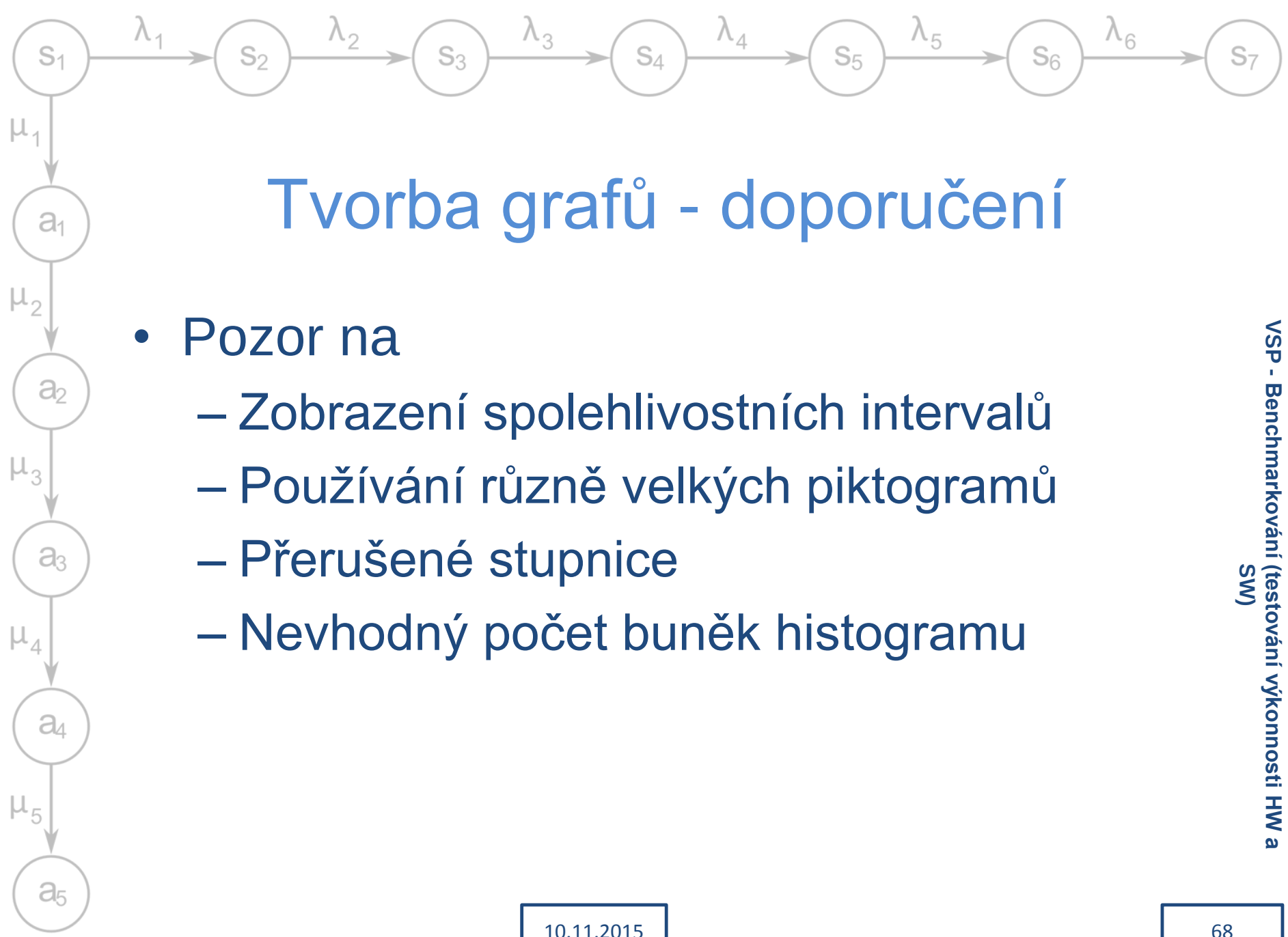
- Nesnažit se znázornit všechno najednou
- Počátek souřadnic v 0 (nebo důsledně označit pokud tomu tak není)
- Nezávislá proměnná na ose x
- Raději text než nesrozumitelné symboly (jako jsou řecká písmena)

Švindlování s počátkem os a měřítkem grafu



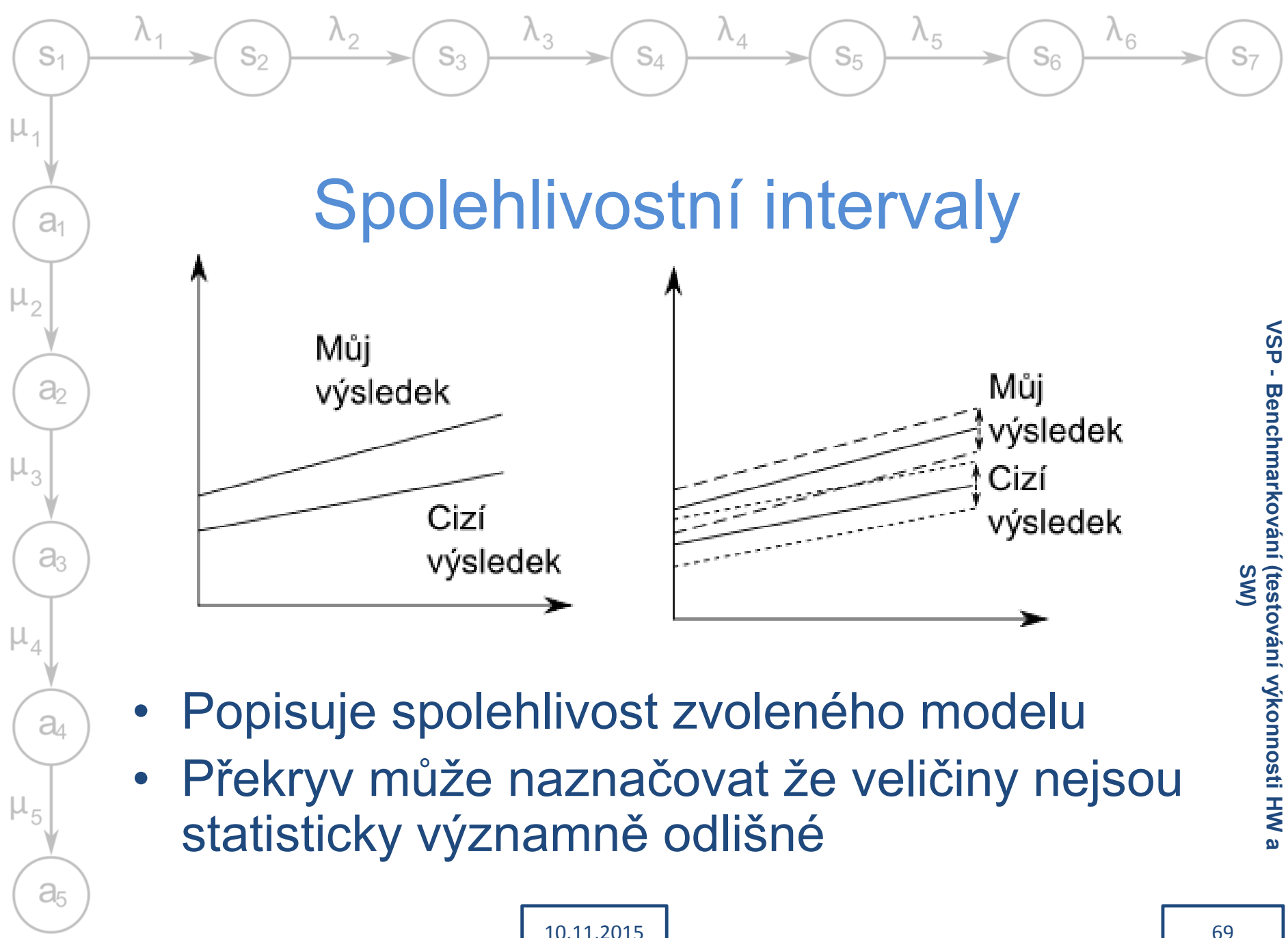


- Můj výsledek je mnohem lepší
- Výsledky jsou téměř stejné



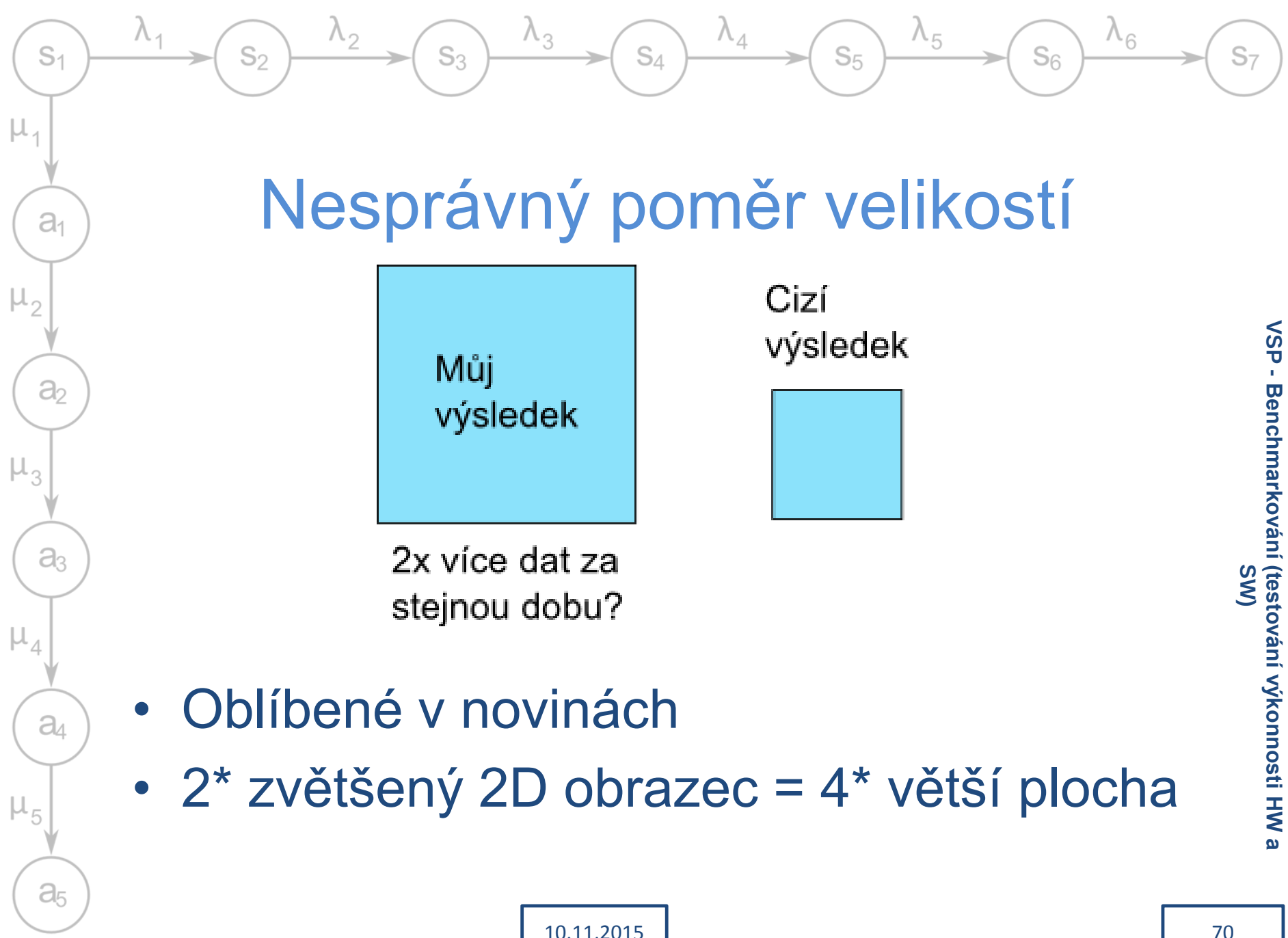
Tvorba grafů - doporučení

- Pozor na
  - Zobrazení spolehlivostních intervalů
  - Používání různě velkých piktogramů
  - Přerušené stupnice
  - Nevhodný počet buněk histogramu



Spolehlivostní intervaly

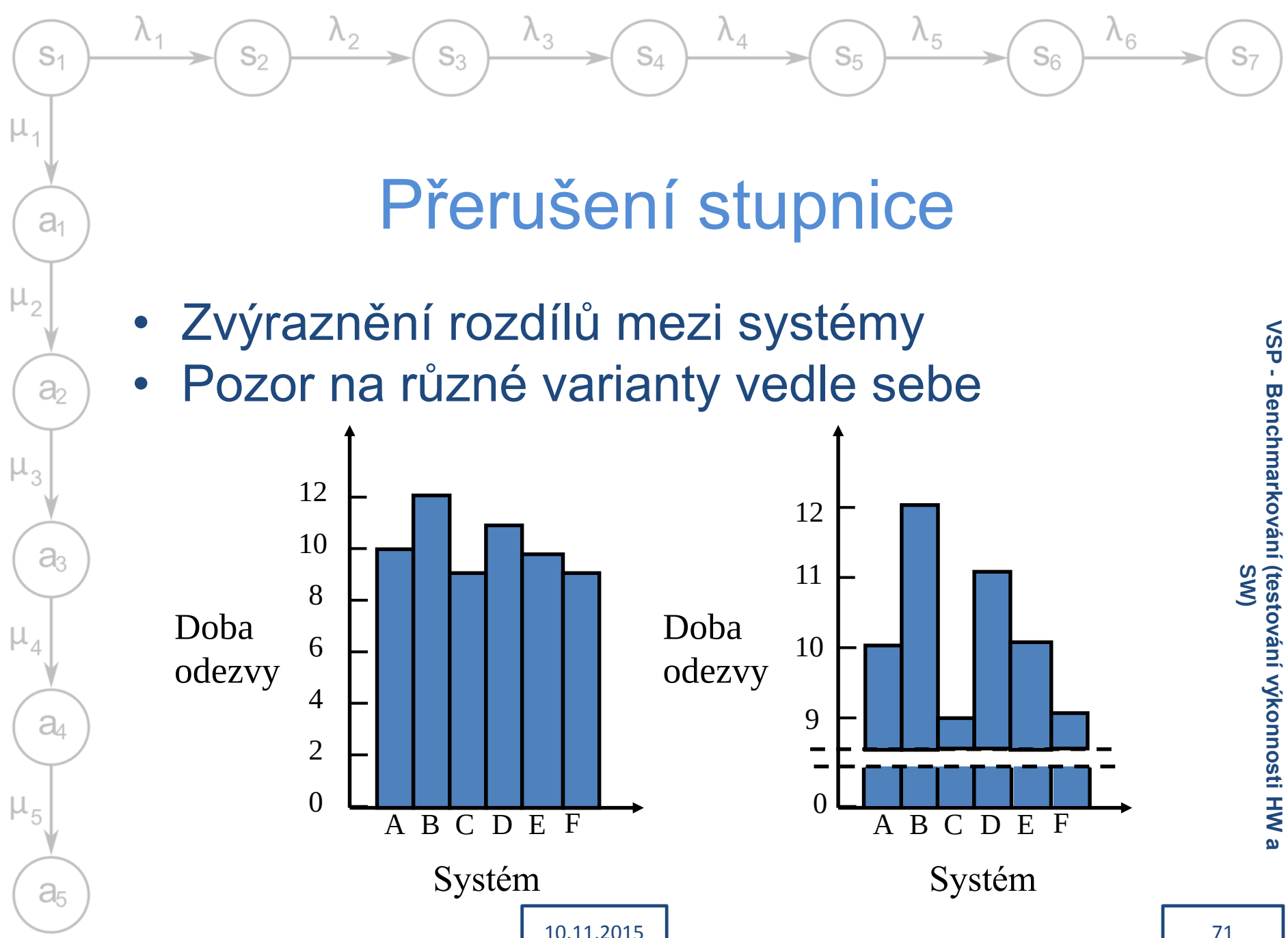
- Popisuje spolehlivost zvoleného modelu
- Překryv může naznačovat že veličiny nejsou statisticky významně odlišné



Nesprávný poměr velikostí

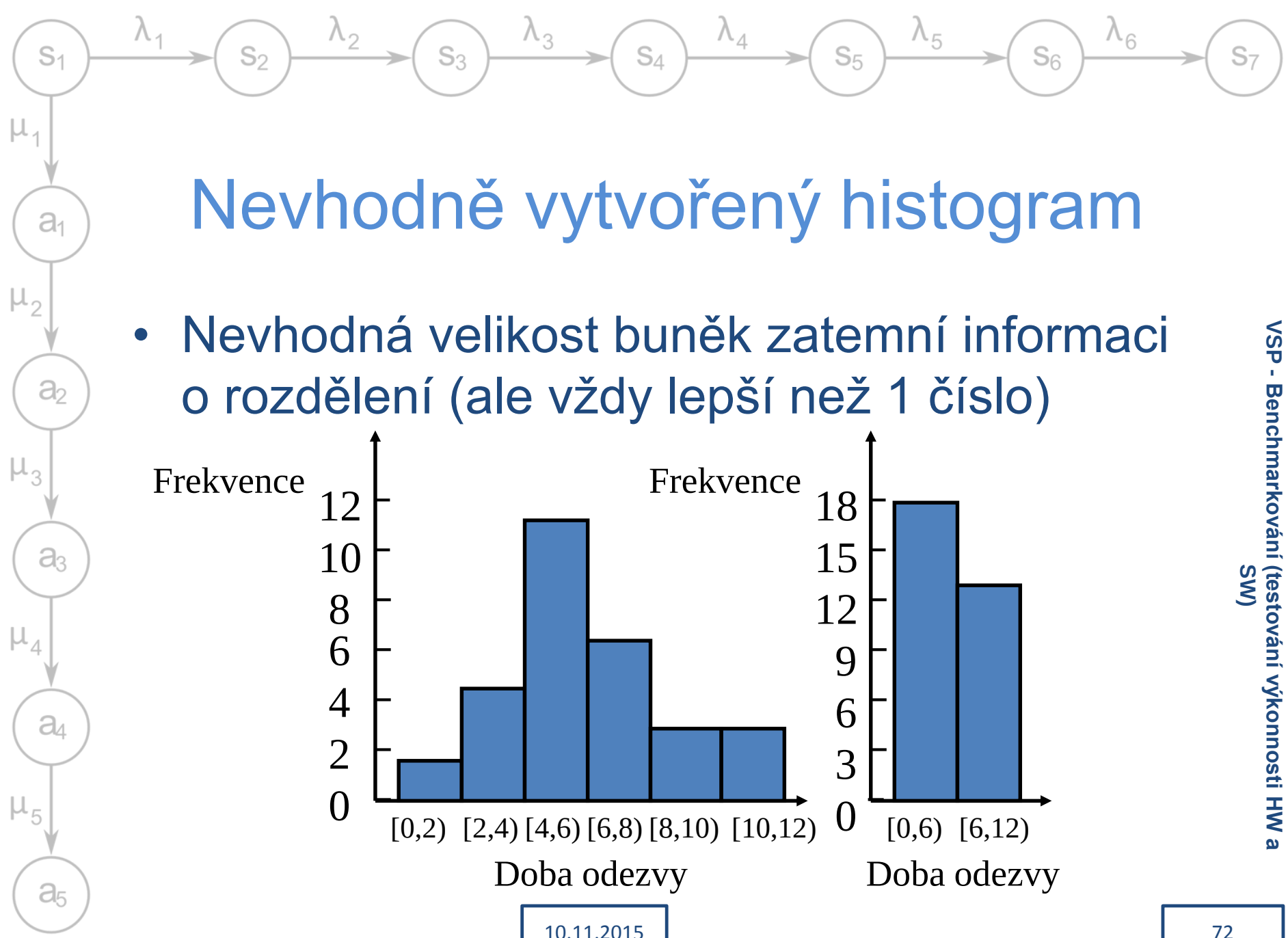
2x více dat za stejnou dobu?

- Oblíbené v novinách
- 2* zvětšený 2D obrazec = 4* větší plocha



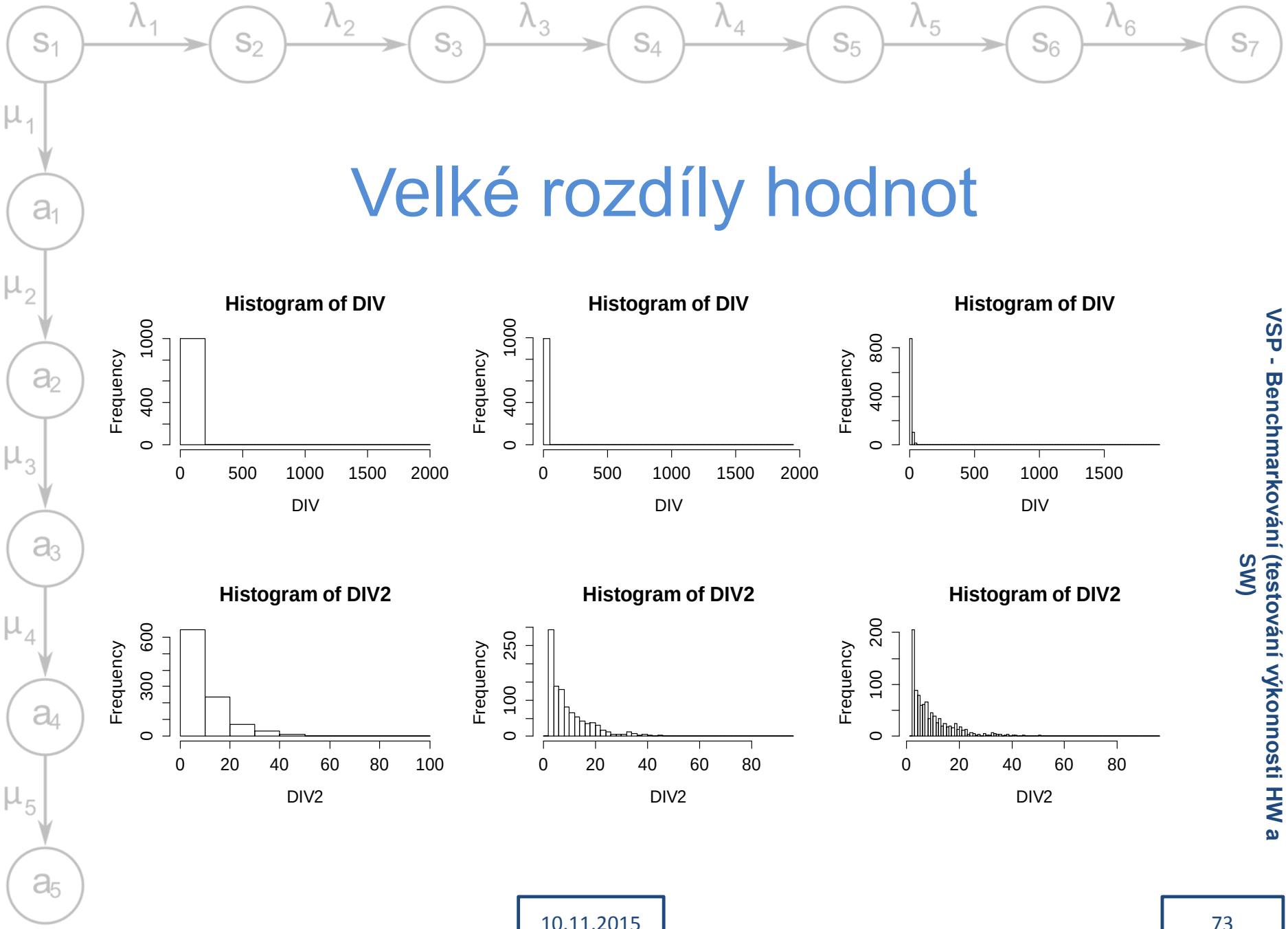
Přerušení stupnice

- Zvýraznění rozdílů mezi systémy
- Pozor na různé varianty vedle sebe

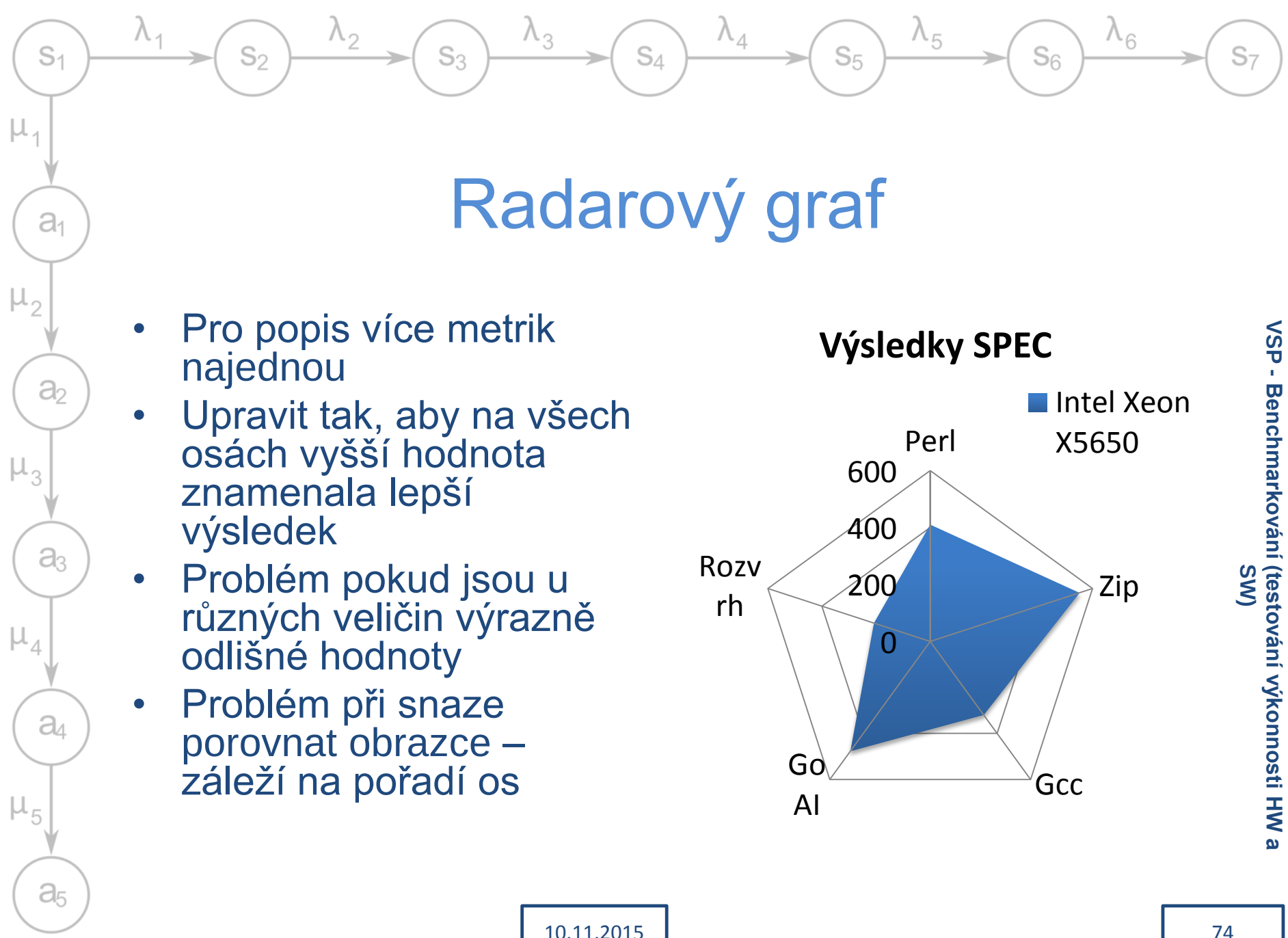


Nevhodně vytvořený histogram

- Nevhodná velikost buněk zatemní informaci o rozdělení (ale vždy lepší než 1 číslo)



Velké rozdíly hodnot



Radarový graf

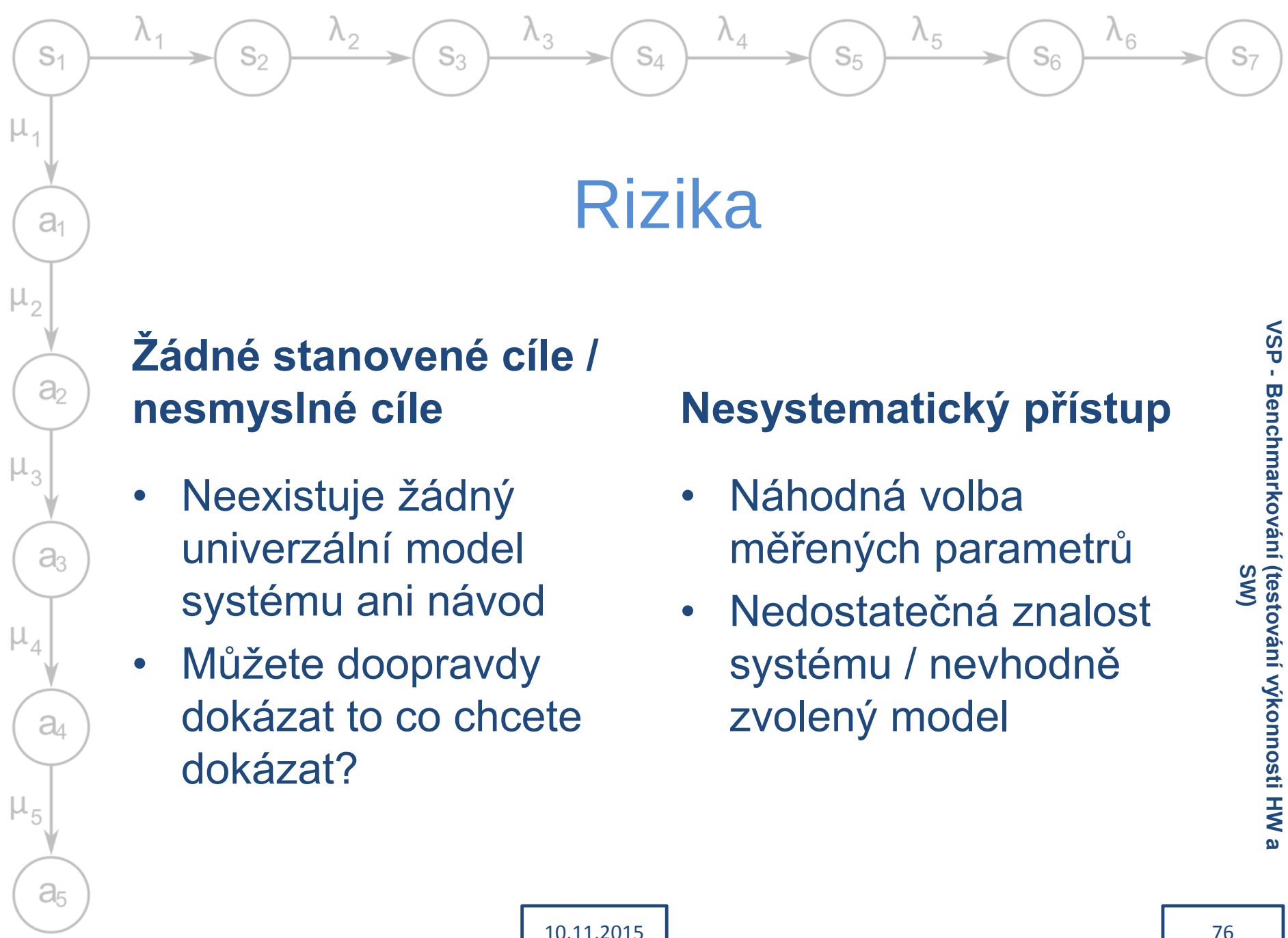
- Pro popis více metrik najednou
- Upravit tak, aby na všech osách vyšší hodnota znamenala lepší výsledek
- Problém pokud jsou u různých veličin výrazně odlišné hodnoty
- Problém při snaze porovnat obrazce – záleží na pořadí os

Příklad – benchmarkování JVM

- Test různých implementací JRE s podporou ARM architektury
- Pro embedded systém bez GUI

- IBM Java J9
- Orace Embedded Java
- IcedTea OpenJDK
- Cacao OpenJDK
- Jamvm OpenJDK
- Zero OpenJDK

- Seralizace
- Databáze Derby
- Šifrování (aes, rsa, podpis)
- Komprese
- Xml (transformace, validace)



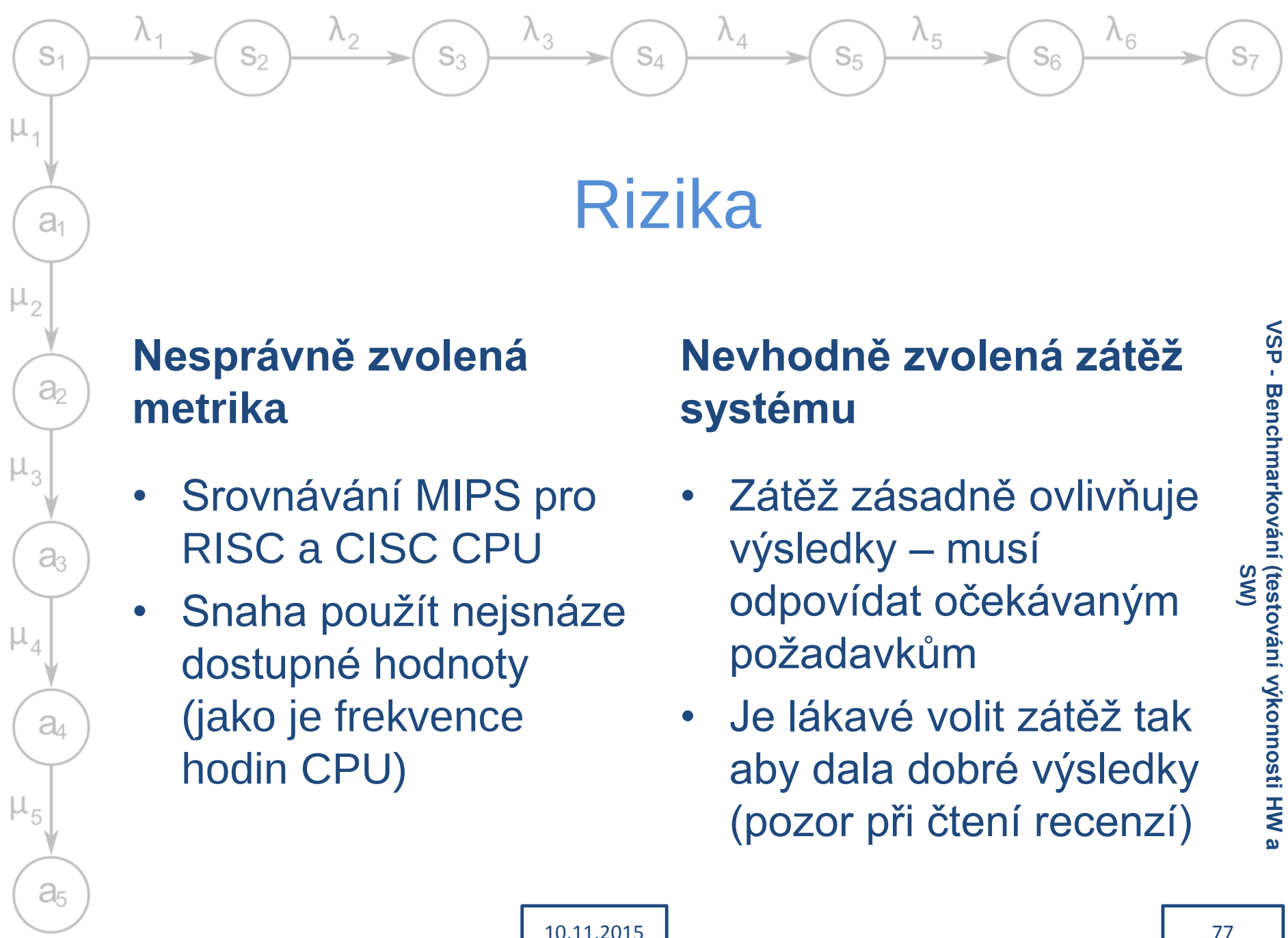
Rizika

Žádné stanovené cíle / nesmyslné cíle

- Neexistuje žádný univerzální model systému ani návod
- Můžete doopravdy dokázat to co chcete dokázat?

Nesystematický přístup

- Náhodná volba měřených parametrů
- Nedostatečná znalost systému / nevhodně zvolený model



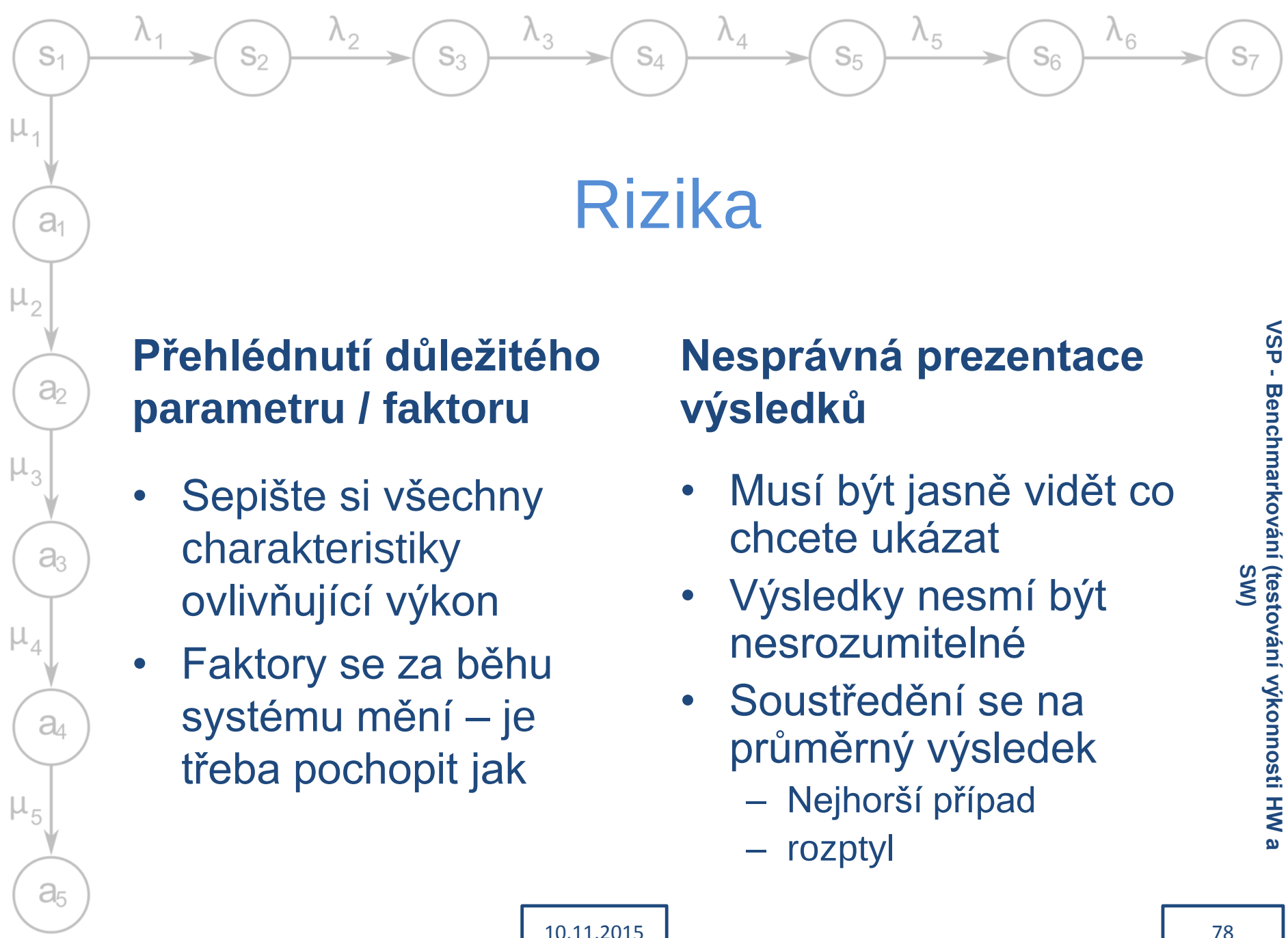
Rizika

Nesprávně zvolená metrika

- Srovnávání MIPS pro RISC a CISC CPU
- Snaha použít nejsnáze dostupné hodnoty (jako je frekvence hodin CPU)

Nevhodně zvolená zátěž systému

- Zátěž zásadně ovlivňuje výsledky – musí odpovídat očekávaným požadavkům
- Je lákavé volit zátěž tak aby dala dobré výsledky (pozor při čtení recenzí)



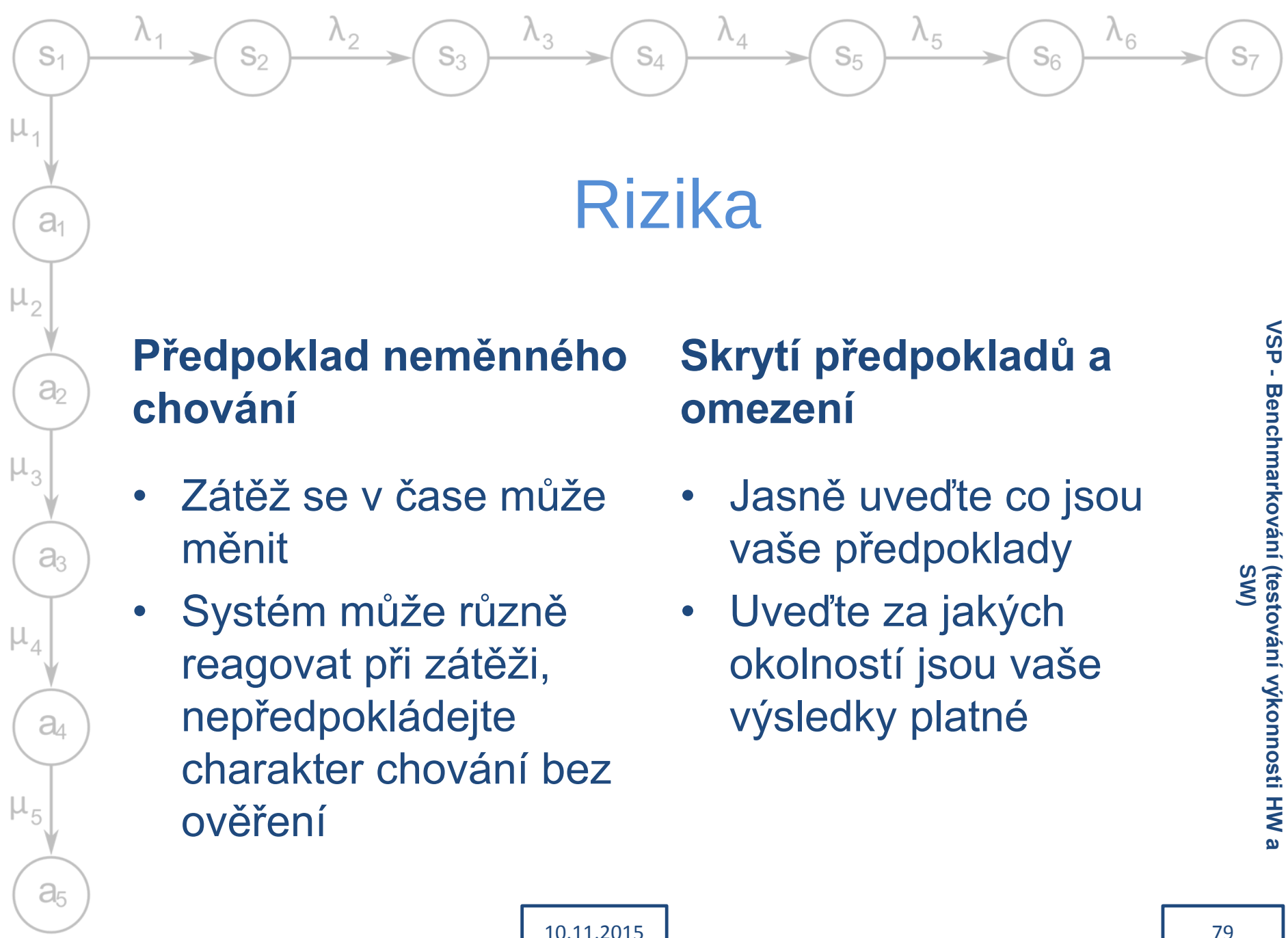
Rizika

Přehlédnutí důležitého parametru / faktoru

- Sepište si všechny charakteristiky ovlivňující výkon
- Faktory se za běhu systému mění – je třeba pochopit jak

Nesprávná prezentace výsledků

- Musí být jasně vidět co chcete ukázat
- Výsledky nesmí být nesrozumitelné
- Soustředění se na průměrný výsledek
  - Nejhorší případ
  - rozptyl



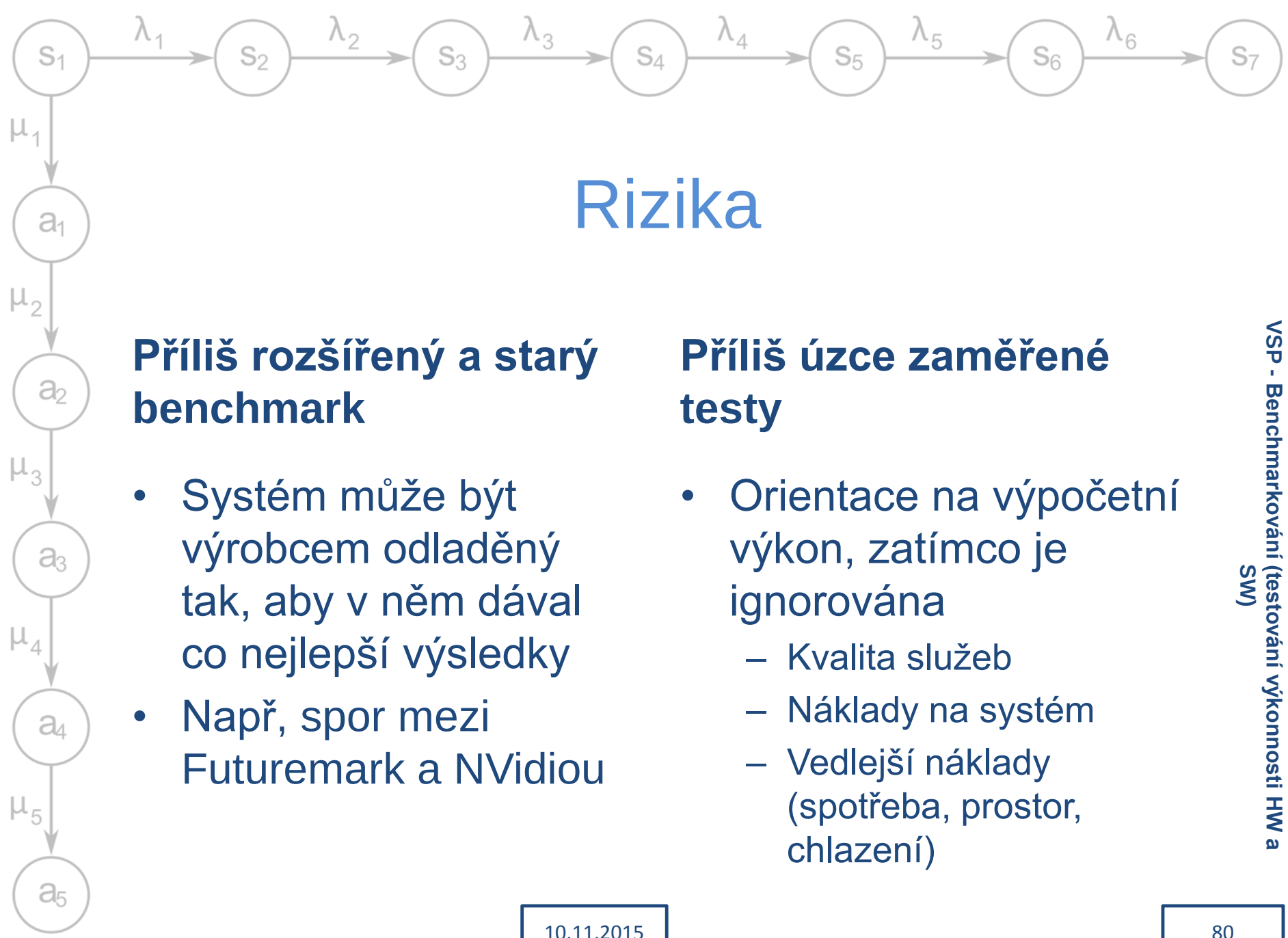
Rizika

Předpoklad neměnného chování

- Zátěž se v čase může měnit
- Systém může různě reagovat při zátěži, nepředpokládejte charakter chování bez ověření

Skrytí předpokladů a omezení

- Jasně uveďte co jsou vaše předpoklady
- Uveďte za jakých okolností jsou vaše výsledky platné



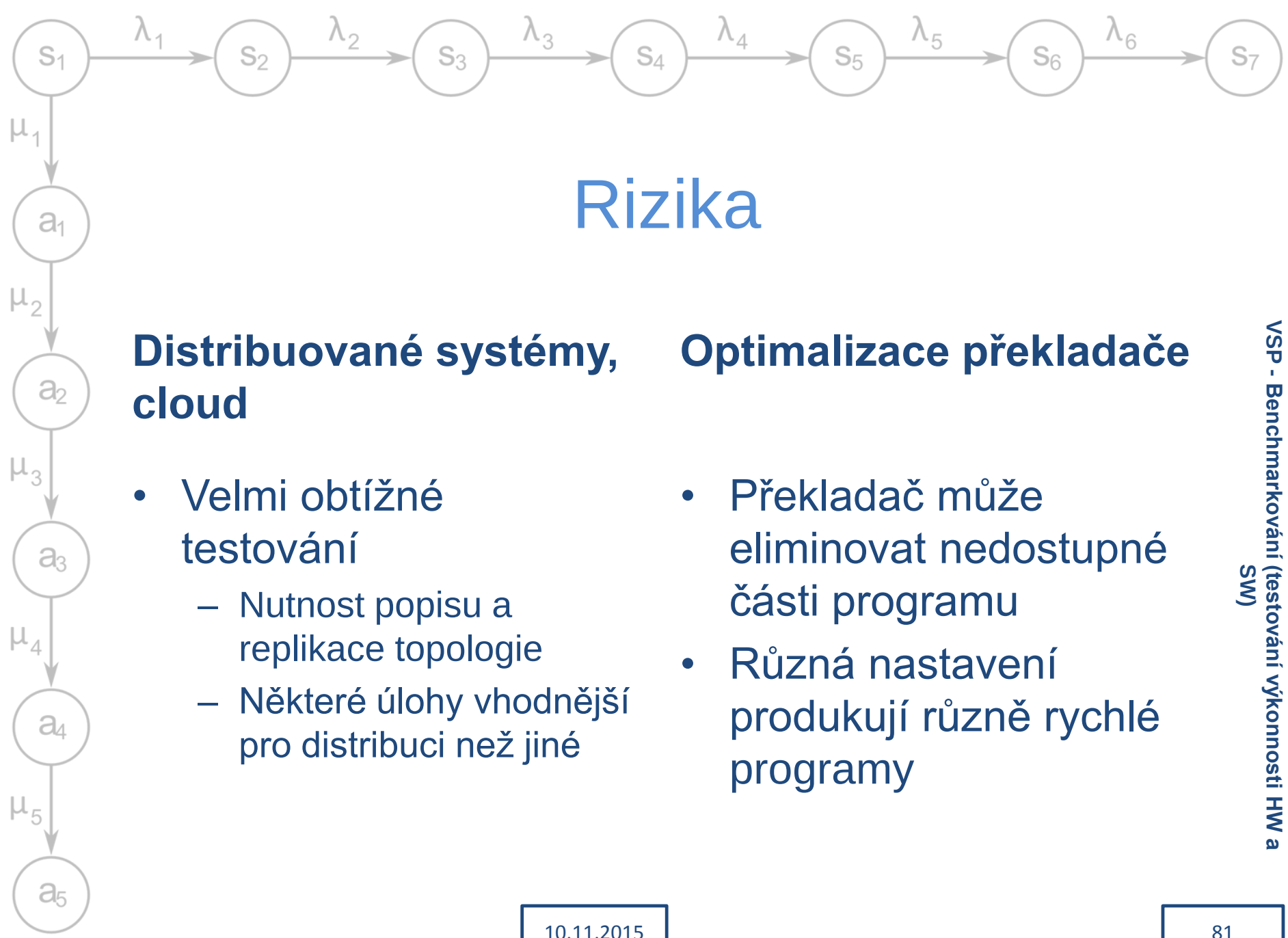
Rizika

Příliš rozšířený a starý benchmark

- Systém může být výrobcem odladěný tak, aby v něm dával co nejlepší výsledky
- Např, spor mezi Futuremark a NVidiou

Příliš úzce zaměřené testy

- Orientace na výpočetní výkon, zatímco je ignorována
  - Kvalita služeb
  - Náklady na systém
  - Vedlejší náklady (spotřeba, prostor, chlazení)



Rizika

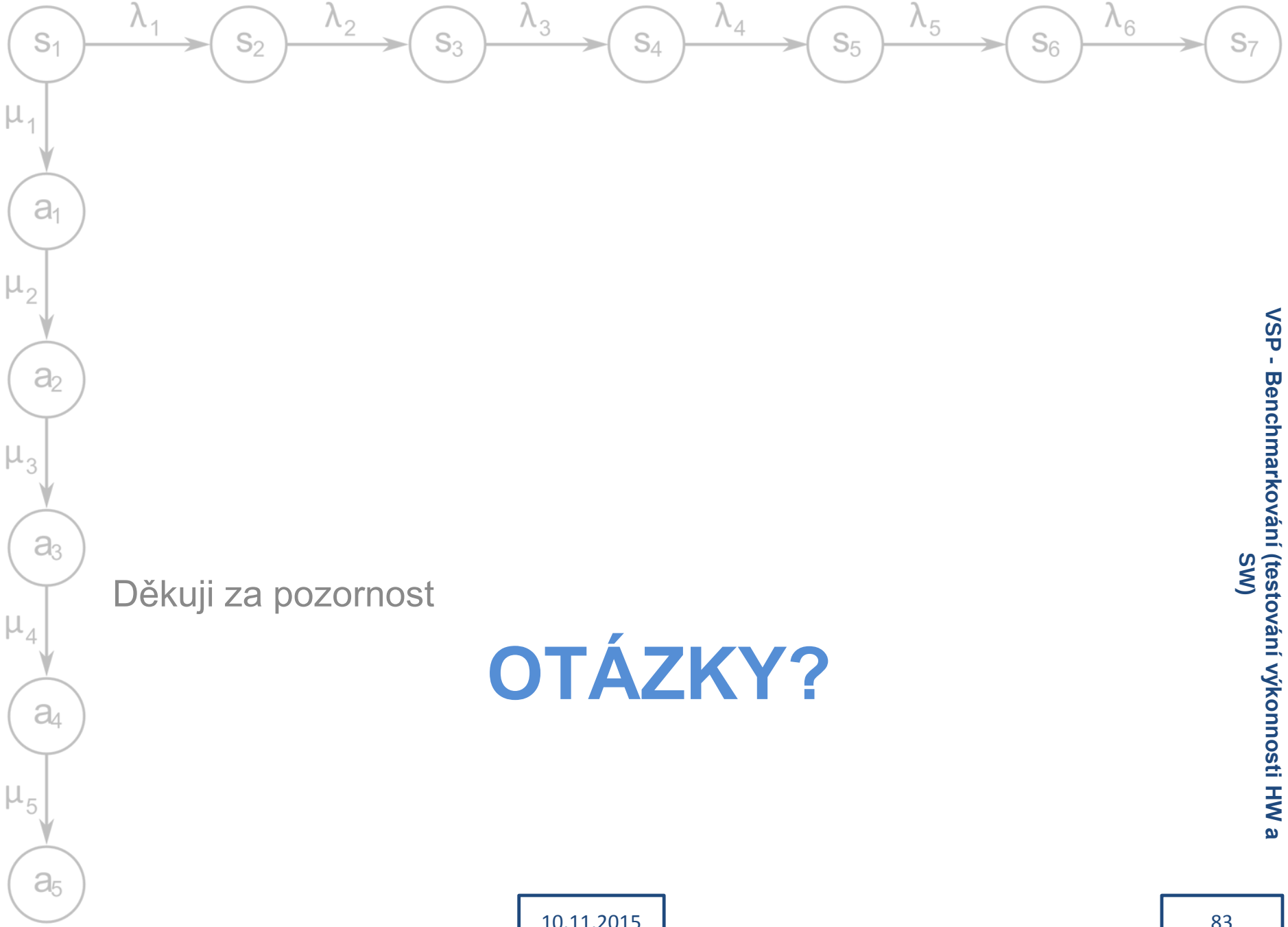
Distribuované systémy, cloud

- Velmi obtížné testování
  - Nutnost popisu a replikace topologie
  - Některé úlohy vhodnější pro distribuci než jiné

Optimalizace překladače

- Překladač může eliminovat nedostupné části programu
- Různá nastavení produkují různě rychlé programy

Rizika – střet s realitou



Děkuji za pozornost

OTÁZKY?