Obsah:

- pojmy z modelování a simulace
- diskrétní systémy a simulátory
- SHO
- konstrukce pro SHO v SIMLIB

# 1 Základní pojmy

- Systém — soubor elementárních částí (prvků systému), které mají mezi sebou určité vztahy
  - reálné systémy
  - nereálné systémy — ještě neexistující, fiktivní
- Model — napodobenina systému jiným systémem
  - reprezentace znalostí
  - typy modelů: fyzikální <> matematické, statické <> dynamické
- Modelování — vytváření modelů systémů
  - modelovat lze jen to, co dokážeme popsat
- Simulace — získávání nových znalostí o systému experimentováním s jeho modelem
- Princip modelování a simulace
  - Realita → Znalosti → Abst. model → Simulační model
  - Cílem je získat nové znalosti o modelovaném systému
- Etapy modelování a simulace
  1. Vytvoření abst. modelu — formování zjednodušeného popisu zkoumaného systému
  2. Vytvoření simulačního modelu — zápis abst. modelu formou programu
  3. Verifikace[1] a validace[2] – ověřování správnosti modelu
  4. Simulace — experimentování se simulačním modelem
  5. Analýza a interpretace dat

## 1.1 Systém

- definice: S = (U, R)
  - U = univerzum, konečná množina prvků systému
  - prvek systému u = (X,Y)
    X je množina všech vstupních proměnných
    Y je množina všech výstupních proměnných
  - R = množina všech propojení prvků univerza

[1] Verifikace — kontroluje převod abs. do sim. modelu
[2] Validace – kontroluje správnost výsledku

- Vazby mezi prvky systému
  - Sériová
  - Paralelní
  - Zpětná vazba

## 1.2 Čas

- Reálný čas — probíhá v něm skutečný děj v reálném čase
- Modelový čas — je časová osa modelu, při simulaci nemusí být synchronní s reálným časem
- Strojový čas — čas CPU spotřebovaný na výpočet programu
- Časová množina — množina časových okamžiků, ve kterých jsou definovány hodnoty vstupních, stavových a výstupních proměnných prvků systému
  - Diskrétní: $T_d = \{1, 2, 3, 4, 5\}$
  - Spojitá: $T_s = \langle 1.0, 5.0 \rangle$

## 1.3 Chování systému

- Každému časovému průběhu vstupní veličiny přiřazuje časový průběh výstupní veličiny
- Dáno vzájemnými interakcemi mezi prvky systému
- Ekvivalence chování systémů:
  - Systémy $S_1$ a $S_2$ považujeme za systémy se stejným chováním, vyvolají-li stejné podněty u obou systémů stejné reakce
- Izomorfní systémy:
  - Systémy $S_1 = (U_1, R_1)$ a $S_2 = (U_2, R_2)$ jsou izomorfní, když a jen když:
    1. Prvky univerza $U_1$ lze vzájemně jednoznačně (1:1) přiřadit prvkům univerza $U_2$
    2. Prvky charakteristiky $R_1$ lze vzájemně jednoznačně přiřadit prvkům charakteristiky $R_2$, a to tak, že prvku charakteristiky $R_1$, vyjadřujícímu orientovaný vztah mezi dvěma prvky univerza $U_1$, je vždy přiřazen právě ten prvek charakteristiky $R_2$, který vyjadřuje stejně orientovaný vztah mezi odpovídající dvojicí prvků univerza $U_2$ a naopak.
  - **Nezahrnuje chování**
- Homomorfní systémy:
  - Systém $S_1 = (U_1, R_1)$ je homomorfní se systémem $S_2 = (U_2, R_2)$ právě když:
    1. Prvkům univerza $U_1$ je možno přiřadit jednoznačně prvky univerza $U_2$ (opačně tomu tak být nemusí, N:1)
    2. Prvkům charakteristiky $R_1$ je možno jednoznačně přiřadit prvky charakteristiky $R_2$, a to tak, že prvku charakteristiky $R_1$ vyjadřujícímu orientovaný vztah mezi dvěma prvky univerza $U_1$ je vždy přiřazen právě ten prvek charakteristiky $R_2$, který vyjadřuje stejně orientovaný vztah mezi odpovídající dvojicí prvků univerza $U_2$ ve smyslu bodu 1.
- Okolí systému:

  - Podstatné okolí systému = vše co má vliv na chování systému a není jeho součástí
    * Uzavřený systém — nekomunikuje s okolím
    * Otevřený systém — komunikuje s okolím

- Klasifikace prvků systémů:
  - Klasifikace 1:
    * Prvky se spojitým chováním
    * Prvky s diskrétním chováním
  - Klasifikace 2:
    * Prvky s deterministickým[3] chováním
    * Prvky s nedeterministickým chováním

- Klasifikace systémů:
  - Spojité — všechny prvky mají spojité chování
  - Diskrétní — všechny prvky mají diskrétní chování
  - Kombinované — obsahuje spojité i diskrétní prvky
  - Deterministické — všechny prvky jsou deterministické
  - Nedeterministické — alespoň jeden prvek s nedeterministickým chováním

# 2 Simulace

= experimentování s reprezentací simulačního modelu

**Cíl simulace:** získání nových informací o chování systému v závislosti na vstupních veličinách a na hodnotách parametrů

**Postup:** opakované řešení modelu (provádění simulačních metod)

- Nastavení hodnot parametrů a počátečního stavu modelu
- Zadávání vstupních podnětů z okolí při simulaci
- Vyhodnocení výstupních dat (informaci o chování systému)

**Typy simulace:** rozdělujeme podle použitého popisu modelu na:

- Spojitá / Diskrétní / Kombinovaná
- Kvalitativní / Kvantitativní

A podle simulátoru na:

- Na analogovém / číslicovém počítači, fyzikální
- "Real-time"simulace
- Paralelní a distribuovaná simulace

[3] Za stejných vstupních podmínek vytvoří stejné výsledky – je předvídatelný

**Zpracování výsledků simulace:**

- Záznam průběhu simulace
- Vizualizace výsledků, animace

Analýza získaných výsledků:

- Intuitivní vyhodnocení, heuristiky, atd.
- Statistické zpracování
- Automatické vyhodnocení (např. pro optimalizaci)
- Provnávání s naměřovanými daty

**Verifikace modelu:** ověřuje korespondenci abstraktního a simulačního modelu, tj. izomorfní vztah mezi AM a SM

**Validace modelu:** proces nad simulačním modelem, v němž se snažíme dokázat, že skutečně pracujeme s modelem adekvátním modelovanému systému

- Jeden z nejobtížnějších problémů modelování
- Vyžaduje neustálou konfrontaci informací, které o modelovaném systému máme a které simulací získáváme
- Nelze absolutně dokázat přesnost modelu
- Pokud chování neodpovídá, musí se model modifikovat

**Simulační nástroje:** Simulační systémy usnadňují vytváření modelů a provádění experimentů. Použité pro:

- práce s abstraktními modely
- programování simulačních modelů
- experimentování se simulačními modely
- vizualizaci a vyhodnocení výsledků

# 3 Modely a modelování

Způsoby popisu modelů (matematicky):

- Konečný automat
- Petriho síť
- Turingův stroj
- atd.

**Specifické cíle a účely modelů:**

- Studium chování systému pro určitá specifická kritéria, zkoumání povahy závislostí mezi parametry a odezvou systému
- *Predikce* — vyhodnocení chování systému za určitých podmínek
- *Analýza citlivosti* — určení faktorů (parametrů), které jsou pro činnost systému nejvýznamnější
- *Optimalizace* — nalezení takové kombinace parametrů, která vede k nejlepší odezvě systému

**Simulační model:** abstraktní model zapsaný formou programu
Vztahy mezi modely:

- **homomorfní vztah:** modelovaný systém — abstraktní model
- **izomorfní vztah:** abstraktní model — simulační model

### 3.0.1 Klasifikace podle Fishwick

**Konceptuální modely:**

- Komponenty nebyly přesně popsány ve smyslu teorie systémů
- Obvykle se používají v počáteční fázi modelování pro ujasnění souvislostí a komunikaci v týmu
- Mají formu textu nebo obrázků

**Deklarativní modely:**

- Popis přechodů mezi stavy systému
- Model je definován stavy a událostmi, které způsobí přechod z jednoho stavu do druhého za jistých podmínek
- Vhodné především pro diskrétní modely
- Obvykle zapouzdřeny do objektů
- Např.: Konečené automaty, Petriho sítě, atd.

**Funkcionální modely:**

- Grafy zobrazující funkce a proměnné
- 2 modifikace: uzel grafu je funkce nebo proměnná
- Např.: SHO, bloková schémata

**Modely popsané rovnicemi (constraint):**

- Rovnice — algebraické, diferenciální, diferenční
- Neorientované grafy
- Např.: Balistika, kyvadlo

**Prostorové modely:**

- Rozdělují systém na prostorově menší ohraničené podsystémy
- Např.: Parciální diferenciální rovnice, celulární automaty

**Multimodely:**

- Modely složené z různých typů modelů, které jsou obvykle heterogenní
- Např.: Kombinované modely, modely s neurčitostí, modely na různé úrovni abstrakce
- **Většina netriviálních modelů spadá do této kategorie**

## 3.1 Simulační nástroje

Možnosti:

- Použití obecných jazyků (C, C++, Java)
- Simulační knihovny pro obecné jazyky (SIMLIB/C++)
- Simulační jazyky
- Simulační systémy
- Propojování různých nástrojů

**Simulační jazyky** = speciální programovací jazyky Poskytují prostředky usnadňující efektivní popis:

- struktury modelů
- chování modelů
- simulačních experimentů

Výhody a nevýhody:

+ Jednodušší popis modelu (snadnější verifikace)
+ Možnost automatické kontroly popisu modelu
– Další jazyk = překladač, výuka, údržba
– Málo používáno = problémy (cena, chyby)

# 4 Diskrétní simulace

**Formy zápisu diskrétních systémů**

- V podobě programu zapsaného v (simulačním) jazyce
- Petriho sítě
- Automaty, sítě automatů

**Procesy:**

- Posloupnost událostí
- Paralelní procesy – současně prováděné procesy
- Kvaziparalelismus – provádění paralelních procesů na jednoprocesorovém počítači

**Paralelismus**

- Popis jednotlivých procesů sekvencí kroků
- Popis komunikace procesů – zprávy (synchronní, asynchronní)
- Synchronizace při používání sdílených prostředků

## 4.1 Petriho sítě

Definice: $\Sigma = (P, T, F, W, C, M_0)$

- P je množina míst (stavy)
- T je množina přechodů, $P \cap T = \emptyset$
- Incidenční relace $F \subseteq (P \times T) \cup (T \times P)$
- Váhová funkce $W : F \to \{1, 2, \ldots\}$
- Kapacity míst $C : P \to N$
- Počáteční značení $M_0 : P \to N$ ($M$ se nazývá značení Petriho sítě)

**Graf Petriho sítě**

- Místa = kružnice
- Přechody = obdélníky
- Incidenční relace = šipky (orientované hrany)
- Váhová funkce = ohodnocení hran

**Proveditelnost přechodu v Petriho síti:** přechod je proveditelný při značení $M$, jestliže:

- ve vstupních místech čeká dostatek procesů
- a současně výstupní místa mají dostatečně volnou kapacitu

**Petriho sítě v modelování** mohou modelovat:

- paralelismus procesů
- komunikaci a synchronizaci procesů
- nedeterminismus

**Prioritní přechody**

- Je-li více přechodů proveditelných z jednoho značení, můžeme jim dát priority
- Pouze nezáporná čísla, implicitně 0
- Vyšší číslo ⇒ vyšší priorita

**Pravděpodobnost provedení přechodu**

- V případě, že jsou přechody různě pravděpodobné
- Musí být zastoupeny všechny zbývající (nelze dát jen 30 % a 70 % už ne)

**Pravidla používání rozšířených přechodů**

- Přechod má pouze jeden parametr (priorita nebo pravděpodobnost nebo časování)
- Není parametrem hrany

**Časované Petriho sítě**

- Konstantní čas čekání
- Náhodně generovaná doba čekání
- Sémantika časovaného přechodu:
  1. Pokud je přechod v čase $t$ proveditelný, spustí se odpočet času
  2. Po celou dobu odpočítávání se nemění stav značek
  3. Na konci doby se provede přemístění značek

# 5 Systémy hromadné obsluhy

**SHO (Queueing systems):** jsou systémy obsahující zařízení (s frontami), která poskytují obsluhu transakcím.

Typický SHO obsahuje:

- Transakce (=procesy) a popis jejich příchodů
- Obslužné linky (více typů) a popis obsluhy
- Fronty různých typů, ve kterých transakce čekají

Co sledujeme při simulaci:

- Informace o čase stráveném transakcí v systému
- Doby čekání ve frontách
- Vytížení obslužných linek

**Vstupní tok požadavků** obvykle jde o stochastický proces příchodů do systému. Kde při modelování příchodů zadáváme:

- Střední dobu mezi příchody (obvykle používáme exponenciální rozložení)
- Počet příchodů za jednotku času (obvykle Poissonovo rozložení)

**Fronty čekajících požadavků** se vytvoří vždy, když požadavek chce být obsloužen již obsazeným zařízením. Pro fronty je charakteristické:

- Řazení požadavků ve frontě (např. FIFO)
- Způsob výběru požadavků z fronty
- Největší možná délka fronty

**Frontové řady:** FIFO, LIFO, SIRO (Service In Random Order)
**Nulová fronta:** požadavek nemůže vstoupit do fronty, jde o systém se ztrátami
**Fronta konečná:** omezení kapacity fronty
**Fronta s netrpělivými požadavky:** netrpělivý požadavek opouští systém, překročí-li doba čekání určitou mez (time-out)

**Prioritní požadavky, priorita obsluhy:**

- Přicházející požadavky nejsou rovnocenné, každý požadavek může mít svoji prioritu
- Prioritních úrovní může být více
- U jedné obslužné linky lze vytvářet i několik front s různými prioritami
- Vstupem požadavku s vyšší prioritou nastane jedna ze čtyř možností pro právě probíhající obsluhu požadavku s nižší prioritou

**Prioritní obsluha:**

1. Započatá obsluha se normálně ukončí (slabá priorita)
2. Obsluha se přeruší a začne obsluha požadavku s vyšší prioritou obsluhy (silná priorita). Požadavek, jehož obsluha byla přerušena
   - Odchází ze systému neobsloužen
   - Vrací se znovu do fronty a když je později znovu obsluhován
     - Obsluha pokračuje od přerušeného místa
     - Začne znovu od začátku
3. Jsou-li všechny linky obsazeny a u každé je fronta, požadavek se sám rozhodne, do které fronty se zařadí
4. Vytvářejí-li požadavky jednu společnou frontu, požadavek vstupuje do té obslužné linky, která se nejdříve uvolní

**Obslužná síť** vznikne spojením několika obslužných linek

- **Otevřená obslužná síť:** výměna požadavků mezi sítí a okolím
- **Uzavřená obslužná síť:** nedochází k výměně požadavků mezi sítí a okolím
- **Smíšená obslužná síť:** pro některé typy požadavků je síť otevřená, pro jiné uzavřená

## 5.1 Kendallova klasifikace SHO

Standard stručného a přehledného vyjádření typu SHO – používá 3 hlavní hlediska:

- **X** – typ stochastického procesu popisujícího příchod požadavku k obsluze
- **Y** – zákon rozložení délky obsluhy
- **c** – počet dostupných linek

Specifikace má tvar **X/Y/c**, kde:

- X, Y – velká písmena $M, D, G, E_n, K_n, Gl$
- c – přirozené číslo, včetně $\infty$

Popis značek:

| Symbol | X | Y |
|---|---|---|
| M | Poissonův proces příchodů, tj. exponenciální rozložení vzájemně nezávislých intervalů mezi příchody | Exponenciální rozložení doby obsluhy |
| $E_k$ | Erlangovo rozložení intervalů mezi příchody s parametry $\lambda$ a $k$ | Erlangovo rozložení doby obsluhy s parametry $\lambda$ a $k$ |
| $K_n$ | Rozložení $\chi^2$ intervalů mezi příchody, $n$ stupňů volnosti | Rozložení $\chi^2$ doby obsluhy |
| D | Pravidelné deterministické příchody | Konstantní doba obsluhy |
| G | Žádné předpoklady o procesu příchodu | Jakékoliv rozložení doby obsluhy |
| Gl | Rekurentní proces příchodů | |

## 5.2 Modelování SHO

Při modelování SHO popisujeme:

- Procesy (transakce) v systému (příchod procesu do systému, jeho činnost, odchod)
- Stav obslužných linek a front u zařízení
- Průběh obsluhy transakcí v zařízeních

### Typy obslužných linek

- $Kapacita = 1$ – Zařízení (Facility)
- $Kapacita > 1$ – Sklad (Store)

# 6 Knihovna SIMLIB

**Prostředky pro diskrétní modelování:**

- **Process** — báze pro modelování procesů
- **Empty** — báze pro modely událostí
- **Facility** — obslužná linka – výlučný přístup
- **Store** — obslužná linka s kapacitou
- **Queue** — fronta
- **Statistiky** — sada tříd pro sběr a uchování statistik

**Modelový čas** reprezentován proměnnou `double Time;` Do proměnné `Time` nelze přiřazovat, překladač vyvolá chybu. Posun času řídí výhradně jádro simulátoru. `Init(t0,t1);` nastaví počáteční čas na `t0`.

**Generátory pseudonáhodných čísel**

- `double Random();)` – rovnoměrné rozložení R(0,1)
- `double Uniform(double L, double H);` – rovnoměrné rozložení, R(L,H)
- `double Exponential(double E);` – exponenciální rozložení se středem E
- `double Normal(double M, double S);` – normální rozložení se středem M a rozptylem S

## 6.1 Události v SIMLIB

Událost je jednorázová (nepřerušitelná) akce vyvolaná v určitém modelovém čase.
Často jsou nutné periodické události – událost se sama znovu naplánuje

```
class Udalost : public Event {
    void Behavior() {
        // akce udalosti
        Activate(Time + e); // muze se periodicky aktivovat
    }
};

// volani
(new Udalost)->Activate();          // na cas Time
(new Udalost)->Activate(Time + e);  // na cas Time + e
```

## 6.2 Procesy

Procesy (transakce) jsou odvozeny z abstraktní třídy `Process`.

```
class Transakce : public Process {
    public:
        Transakce( parametry ) { // konstruktor
            // nepovinny popis inicializace
        }
        void Behavior() {
            // popis chování procesu
        }
};
```

Po aktivaci procesu se volá metoda `Behavior`. Provádění metody je přerušeno při čekání:

- Ve frontě u zařízení (v rámci `Seize`, `Enter`)
- Při explicitním `Wait(dt);` – abstrakce nějaké činnosti trvající dt časových jednotek

Proces se provádí jako posloupnost událostí – např.:

```
void Behavior() {
    ...
    Wait(3);
    ...
}
```

Proces čeká 3 časové jednotky. Při simulaci to znamená, že se naplánuje další jeho pokračování na čas `Time+3` (příkazem `Activate(Time+3)`). Aktivační záznam této události je uložen do kalendáře a proces je přerušen (a spustí se první plánovaná akce v kalendáři).

## 6.3 Kalendář událostí

Kalendář je uspořádaná datová struktura uchovávající aktivační záznamy budoucích událostí.

- Každá naplánovaná budoucí událost (next event) má v kalendáři záznam (acttime, priority, event)
- Kalendář umožňuje výběr prvního záznamu s nejmenším aktivačním časem a vkládání/rušení aktivačních záznamů

## 6.4 Vytvoření, registrování a zrušení procesu

Vytvoření instance třídy: `Transakce *t = new Transakce();`
**Plánování (re)aktivace procesu do kalendáře:**
`t->Activate(tm);` Aktivuje se v čase tm (implicitně je tm = Time).
**Zrušení procesu i jeho registrace ve všech strukturách (fronty, kalendář):**
`t->Cancel(); // nebo delete t;`
**Suspendování běžícího procesu:**
`Passivate();` **Pro události lze použít pouze:** `Activate` a `Cancel`.

# 7 Otázky z minulých let

## 7.1 M/M/1 – zakreslit pomocí PN a demonstrujte SIMLIBem

Systém má exponenciální rozložení intervalů mezi příchody, exponenciální rozložení dob trvání obsluhy a jednu obslužnou linku.

## 7.2 Definujte pojmy model, modelování, modelový čas, simulace, časovaný přechod u stochastické Petriho sítě.

- Model – napodobenina systému jiným systémem
- Modelování – vytváření modelů systémů
- Modelový čas – časová osa modelu, při simulaci nemusí být synchronní s reálným časem
- Simulace – získávání nových znalostí o systému experimentováním s jeho modelem
- Časovaný přechod – pokud je přechod v čase t proveditelný, spustí se odpočet času, na konci doby se provede přemístění značek

## 7.3 Popište strukturu obslužné linky s kapacitou 1 a ve formě pseudokódu zapište operace obsazení a uvolnění linky procesem.

## 7.4 Priorita obsluhy u SHO, demonstujte simlibem a PN.

## 7.5 Uvést pseudokódy seize a release u facility s prioritou obsluhy a uvest, jaké musí mít taková fronta atributy.

## 7.6 Popsat všechny typy priorit používané u SHO a Petriho sítí a každou popsat kódem v SIMLIBu a PN.

## 7.7 Popište části, ze kterých se skládají systémy hromadné obsluhy. Napište pseudokód pro operace zabrání a uvolnění obslužné linky transakcí.

- Transakce (= procesy) a popis jejich příchodů
- Obslužné linky a popis obsluhy
- Fronty různých typů, ve kterých transakce čekají

## 7.8 Popište varianty modelování priorit transakcí na obslužných linkách a zakreslete je pomocí PN