Kapitola 1

Úvod

Tento dokument pojednává o funkčnosti, tvorbě a základních informacích o tvorbě simulátoru modelů P/T[1] Petriho sítí ve formě knihovny k programovacímu jazyku C++. Za pomocí statické a nebo dynamické knihovny je možné vytvořit model Petriho sítě a následně jej nasimulovat. Jedná se stochastické černobílé P/T Petriho sítě s rozšířenými vlastnostmi o pravděpodobnosti, prioritu a časování přechodů, které jim dávají výpočetní sílu Turingova stroje, tedy jsou schopny počítat všechny vyčíslitelné funkce.

1.1 kapitola

Na projektu pracoval dvoučlenný tým složený ze studentů bakalářského studia, úkoly byly rozděleny následovně:

- Dušan Kovačič (vedoucí týmu) - jádro simulace, dokumentace.
- Jaroslav Sendler - kalendář, generátory pseudonáhodných čísel, statistika a výpisy, dokumentace.

Pro účely řešení tohoto projektu nebyl využit žádný pravidelný poradce, tudíž nebylo použito žádných rad z třetích stran až na jednu konzultaci s hodnotícím, panem Hrubým. Odborná a pomocná literatura:

1. Rábová Z. a kol: Modelování a simulace, VUT Brno, 1992, ISBN 80-214-0480-9
2. Soubor materiálů prezentovaných na přednáškách
3. JANOUŠEK, Vladimír: Modelování objektů Petriho sítěmi, Brno, 1998. 137 s. Dizertační práce. VUT, Fakulta elektrotechniky a informatiky

[1] Place/Transitions

Kapitola 2

Rozbor tématu a použitých metod/technologií

Petriho sít je matematická reprezentace diskrétních distribuovaných systémů. Vznikla za účelem modelovat řídící systémy.

Definice 2.1. Základní zápis Petriho sítě je možný v grafické formě popisující stavy a přechody mezi stavy, umožňuje vyjádření paralelismu a nedeterminismu (odkaz na předmět[1], opora IMS strana 31)

Definici v podobě matematického zápisu viz. Definice P/T Petriho sítě (odkaz na předmět, slajd 126).

Definice 2.2. Petriho síť je šestice $\Sigma = (P, T, F, W, C, M_0)$

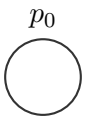
kde:

- P je množina míst
- T je množina přechodů, $P \cap T = 0$
- $F \subseteq (P \times T) \cup (T \times P)$ incidenční relace
- $W : F \to \{1,2, ...\}$ váhová funkce
- Kapacity míst. $C : P \to N$
- M_0 počáteční značení, $M_0 : P \to N$
- (M se nazývá značení Petriho sítě)

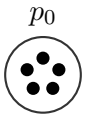
Petriho síť je tvořena pomocí následujících objektů (odkaz na předmět, slajd 127):

Místa

Místa (Places), modelují parciální stavy systému. Jejich vizuální vyjádření v grafu Petriho sítí je ukázáno na obrázku 2.1.



Obrázek 2.1: Vyjádření místa v grafu



Obrázek 2.2: Vyjádření místa s 5 značkami v grafu

Vlastnosti míst jsou definovány pomocí značek (tokens), které vyjadřují okamžitý stav systému. Tento stav je zobrazen na obrázku číslo 2.2.

Je-li značka v místě přítomna, modeluje to skutečnost, že dané stanovisko, uplatňované při posuzování, stavu je momentálně aktuální, nebo-li podmínka je splněna. Každé místo má nadefinovaný počáteční stav (počet značek na začátku) a kapacitu (maximální počet značek na místě). Místo bez ohodnocení je chápáno jako místo s neomezenou kapacitou.

Přechody

Přechod (Transitions), definují vzory možných událostí. Mezi vlastnosti jež definují přechod patří vstupní a výstupní místa, které jsou vyjádřeny orientovanými hranami spojující místa a přechody (viz.2).

Obrázek 2.3 zobrazuje grafický náčrt přechodů. Část A ukazuje zapsání vstupu do přechodu a část B zapsání výstupu.

Obrázek 2.3: A - Vstup do přechodu, B - výstup z přechodu

Přechod je proveditelný pouze pokud splňuje vstupní a následně výstupní podmínky. Jako vstupní podmínky se jeví obsah značek v místě vstupu. Při provádění přechodu se značky ze vstupního místa odeberou a přesunou se na výstupní místo, tímto se použijí výstupní podmínky. Zavedením rozšíření pro Petriho sítě zvětšujeme jejich popisnou sílu. Jsou to:

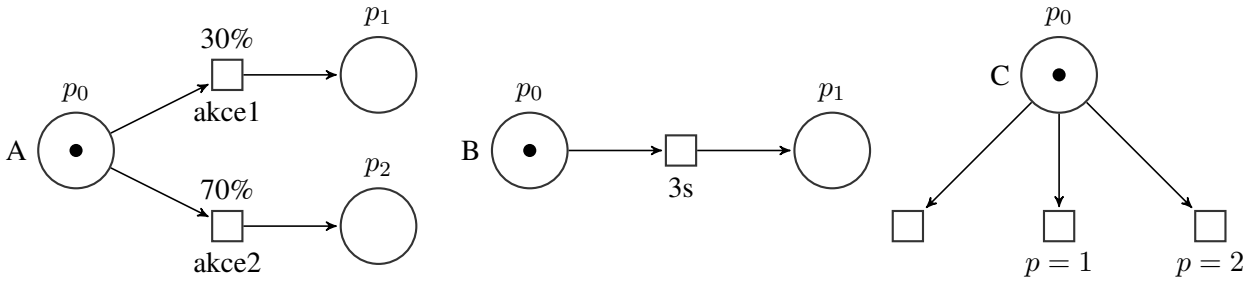
priority přechodů - (odkaz na předmět, slajd 131) je-li přechod s vyšší prioritou proveditelný, tak se provede dřív než kterýkoliv proveditelný přechod s nižší prioritou. Pokud přechodu neurčíme velikost priority jako defaultně je brána hodnota *0*, tato hodnota je i rezervována pro přechody časované. Díky prioritě je možno řešit deterministické konflikty.

pravděpodobnosti přechodů - (odkaz na předmět, slajd 133) je udávána v procentech. Celkový součet pravděpodobnosti všech přechodů ve stejné úrovni musí být *1*, neboli *100%*.

časování přechodů - (odkaz na předmět, slajd 135) do Petriho sítí přináší možnost pracovat s časem. Doposud byly všechny změny v síti provedeny okamžitě. Nyní změny mohou trvat určitou dobu.

Tato rozšíření se navzájem nesmí kombinovat (odkaz na předmět, slajd 134). Obrázek 2.4 ukazuje použití P/T s rozšířeními.

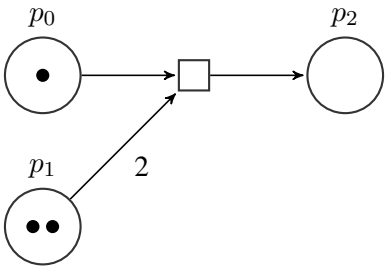
[1] https://www.fit.vutbr.cz/study/courses/index.php?id=7409



Obrázek 2.4: A - pravděpodobnost přechodu, B - časovaný přechod, C - priority přechodů

Orinetované hrany

Hrany (Arcs), spojují místa s přechody nebo přechody s místy. Vlastnost, která udává počet značek, které se současně po hraně přesunou se označuje *váhová funkce* (odkaz na předmět, slajd 127). Defaultně se chápe s hodnotou *1*. Obrázek 2.5 zobrazuje příklad možnosti zápisu.



Obrázek 2.5: Hrana s kapacitou

2.1 kapitola

Simulátor byl navržený tak, aby uspokojoval nejzákladnější parametry využití a byl jednoduchý a srozumitelný pro uživatele. Základní cíle byly tedy stanoveny na jednoduchost a intuitivnost. S trochou představivosti je absolutně předvídatelné co každá zadaná funkce dělá a jaké je jej použití. Na výsledný simulátor by bylo poměrně jednoduché navázat grafické uživatelské rozhraní. Dalším významným kriteriem při tvorbě, byla nutnost eliminovat co nejvíce chyb při samotném zápise zkoumaného modelu. Při prvním pohledu na složitější model, může být jeho význam nejasný, ale po podrobnějším prostudování je koncepčně totožný s grafickým zápisem. Samotný nápad o zápis modelu a jeho ovládání je výhradně dílem autora, inspirovaný podobnou knihovnou SIMLIB (odkaz na předmět, slajd 167).

2.2 kapitola

PetriSim 1.0.0 byl vytvořený jen z poznatků a nápadů autorů, bez ovlivnění studováním funkčnosti jiných podobných nástrojů pro simulaci. Použité algoritmy ne jen, že umožňují zápis a simulaci Petriho sítě s rozšířeními definovanými v úvodě, ale i sémantickou kontrolu správnosti výsledného modelu a schopnost adekvátní simulace. Bez této kontroly by byl například možný nekonečný cyklus v nulovém čase, a nebo průběh nekonečně trvající simulace. V případě takovýchto zjištění, je uživatel okamžitě upozorněný o vzniklé situaci a je mu navrhnuto řešení. Největší motivací pro nekopírování nápověd byla tužba vymyslet něco originálního. Autoři při začátku tvorby již měli zkušenosti s danou problematikou.

Simulátor je postavený na konceptu jednoduchosti a intuitivnosti, aby se co nejvíce přiblížil logikou grafickému zápisu a byl tak lehko pochopitelný pro uživatele. Skládá se z třech hlavních tříd a to `SCTransition`, `SCPlace` a `SCDirectedArc`, při čem každá z nich reprezentuje určitou část grafického zápisu. Třída `SCTransition` přechod, třída `SCPlace` místo a třída `SCDirectedArc` hranu. Každé třídě je možné přidělit nastavení podle definice Petriho sítě. Podobně jako při grafickém zápise, je na vytvoření konečného modelu potřebné pospojovat všechny elementy. Podle definice je možné spojit přechod s místem, na co slouží funkce `TrToPl(přechod,míto,hrana)`, zkratka „TransitionToPlace“, a nebo místo s přechodem pomocí funkce `PlToTr(místo,přechod,hrana)`, zkratka „PlaceToTransition“. Konkrétní funkčnost, přesné možnosti nastavení jednotlivých elementů, jako i nastavení výpisu statistik a provedení simulace budou popsány v dalších kapitolách.

Kapitola 3

Architektura simulačního modelu/simulátoru

Pro samotné použití knihovny je nutné si ve zdrojovém souboru jazyka C++ přidat hlavičkový soubor `PetriSim.h`, který plně zpřístupní danou knihovnu. Na výběr je statická a nebo dynamická knihovna, kterou je nutné přilinkovat při kompilaci výsledného programu.

3.1 Třídy

SCBase

Základní rozhraní pro třídy SCPlace a SCTransition

METODY	
`string` **`GetName()`**	
parametry	žádné
funkčnost	vrátí aktuální jméno objektu
vrácená hodnota	jméno objektu
`void` **`SetName(`**`string name`**`)`**	
parametry	jméno - `string`
funkčnost	nastavení jména objektu
vrácená hodnota	žádná
`int` **`GetStatus()`**	
parametry	žádné
funkčnost	vrátí status objektu
vrácená hodnota	aktuální stav objektu, bližší informace v `StatusList.h`

SCPlace : SCBase

Je náhradou místa v grafické reprezentaci.

METODY	
`int` **`SetArgCapacity(`**`unsigned int capacity`**`)`	
parametry	kapacita - `unsigned int`
funkčnost	nastavení místu maximální kapacitu
vrácená hodnota	aktuální stav místa
`int` **`SetArgStartVal(`**`unsigned int startVal`**`)`	
parametry	počáteční hodnota - `unsigned int`
funkčnost	nastavení počáteční hodnotu značek
vrácená hodnota	aktuální stav místa

SCTransition:SCBase

Je náhradou přechodů v grafické reprezentaci.

METODY	
`int` **`SetArgPrio(`**`unsigned int prio`**`)`	
parametry	priorita - `unsigned int`
funkčnost	nastavení přechodu zadanou prioritu
vrácená hodnota	aktuální stav přechodu
`int` **`SetArgTime(`**`double time, int type = TIME_ABS`**`)`	
parametry	čas - `double`
	typ času - `int`
	`TIME_NORM` - základní čas, absolutní čas
	`TIME_EXP` - exponenciální rozložení
funkčnost	nastaví čas a jeho typ
vrácená hodnota	aktuální stav přechodu
`int` **`SetArgTimeEx(`**`double from, double to, int type = TIME_ROV`**`)`	
parametry	interval čas od - `double`
	interval čas do - `double`
	typ času - `int`
	`TIME_NORM` - základní čas, gaussovo rolzožení
	`TIME_ROV` - základní čas,rovnoměrné rozložení
funkčnost	nastaví čas a jeho typ
vrácená hodnota	aktuální stav přechodu
`int` **`SetArgProbability(`**`double probability`**`)`	
parametry	pravděpodobnost - `double`
funkčnost	nastaví pravděpodobnost dabému přechodu
vrácená hodnota	žádná

SCStats

Třída starající se o statistiky simulace. Výpis statistiky závisí na použití funkcí `PrintSteps(bool stav)` a `PrintSummary(bool stav)`. Počáteční hodnota tisku statistik nachází ve stavu „tiskni“. Pokud je příznak nastaven, statistiky se tisknou na standardní výstup. Funkce `PrintSteps` ovládá tisk výpisů po jednotlivých krocích, tiskne se tedy pokud proběhne v simulaci změna. Druhá funkce `PrintSummary` se stará o tisk souhrnné výsledné statistiky.

Pomocí privátní proměnné třídy `CSPlace` lze měnit tisk statistiky po krocích pro jednotlivá místa. Tato proměnná je pro každé místo nastavena na true, její hodnotu můžeme změnit funkcí `SetArgPrint( bool stav)`.

3.2 Funkce

`void` **`SetSimulationLength(`**`double time`**`)`	
parametry	maximální délka simulace - `double`
funkčnost	nastaví maximální délku běhu simulace
vrácená hodnota	žádná
`int` **`Run()`**	
parametry	žádné
funkčnost	provede celou simulaci, zkontroluje validitu modelu a správný běh simulace
vrácená hodnota	v případě jak nastane chyba s modelem nebo s jeho úspěšným dokončením, upozorní uživatele a vrátí `-1`
	v případě úspěšného provedení simulace vrátí `0`

3.3 Použití knihovny

Použití knihovny je demonstrováno na modelu číslo 4.4.

```
PrintSteps(true)            // nastavení tisku statistiky po krocích
PrintSummary(false)         // vypnutí tisku celkové souhrnné statistiky
SCPlace p;                  // vytvoření místa
SCDirectedArc a;            // vytvoření přechodu
SCTransition t;             // vytvoření šipky (hrany)
t.SetArgTime(5,TIME_EXP);   // nastavení přechodu
TrToPl(&t,&p,&a);           // urření vazby mezi přechodem a místem, orientovaná šipka
SetSimulationLength(100);   // nastavení délky běhu simulace

Run();                      // start simulace
```

3.4 Makefile

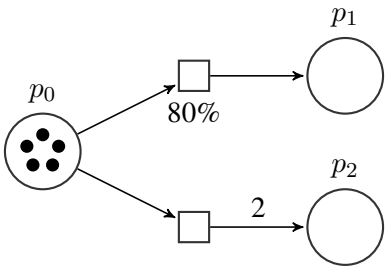
Kapitola 4

Podstata simulačních experimentů a jejich průběh

Model byl vytvořen na základě dokázání správnosti našeho analyzátoru. Experimentováním s různými primitivními modely byla zkontrolována Korektnost simulátoru.

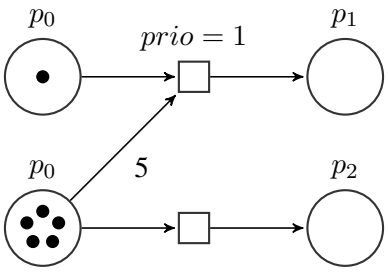
4.1 Primitivní modely

Obrázek 4.1 reprezentuje funkčnost pravděpodobnosti přechodů.



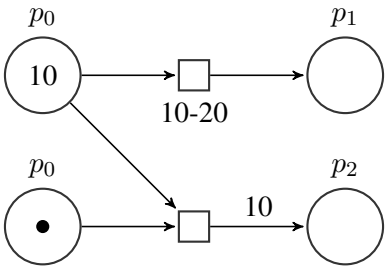
Obrázek 4.1: Model reprezentující procenta přechodů

Obrázek 4.2 reprezentuje funkčnost priorit přechodů.



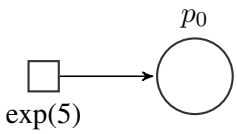
Obrázek 4.2: Model reprezentující prioritu přechodů

Obrázek 4.3 reprezentuje funkčnost časování přechodů.



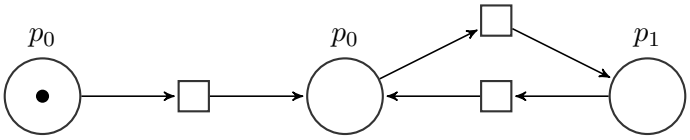
Obrázek 4.3: Model reprezentující časovaný přechod

Obrázek 4.4 reprezentuje funkčnost generování značek s exponenciálním rozložením 5.



Obrázek 4.4: Model reprezentující časovaný přechod

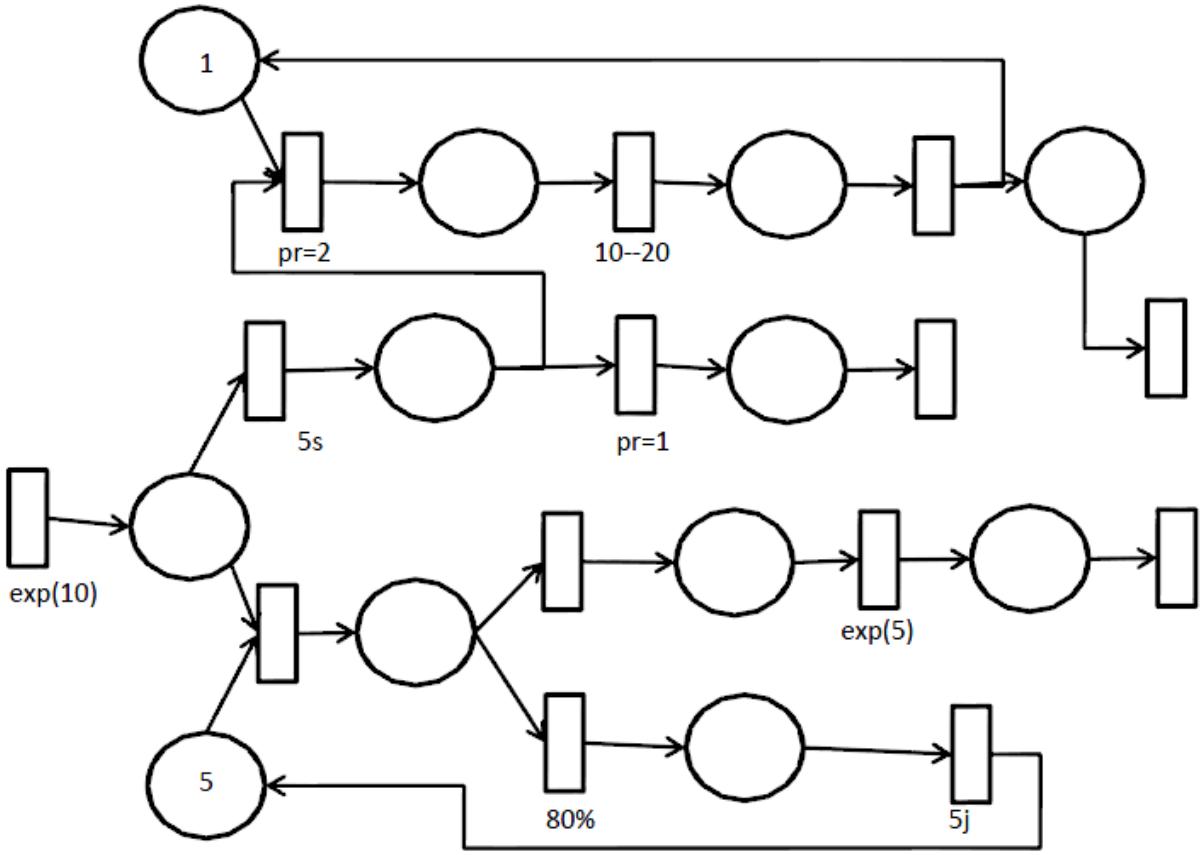
Obrázek 4.5 reprezentuje funkčnost nekonečné smyčky.



Obrázek 4.5: Model reprezentující nekonečnou smyčku

4.2 Model

Na obrázku 4.6 je vyobrazen komplexní model Petriho sítě. Tato síť byla vymyšlena za účelem ukázat možnosti implementované knihovny. Tudíž nevychází z reality. Spojuje zde všechny požadované vlastnosti míst, přechodů i hran.



Obrázek 4.6: Model reprezentující všechnu funkčnost knihovny

4.3 Statistika

Příklad statistiky po krocích je zobrazen na obrázku 4.7. Každý řádek začíná parametrem `time` určující aktuální čas. Pomocí šipek `-->`(zvětšení kap.) nebo `<--`(zmenšení kap.) je graficky zobrazena změna kapacity místa. Část `Place 0 ...` zobrazuje jméno místa, následuje informace o změně kapacity, o váze hrany, přes kterou se proces provedl a o jméně přechodu.

```
time:'2.3' -->Place 0(Act. value ['0'-->'1'])--weight('1')--> Transition 0
time:'2.4' <--Place 0(Act. value ['1'-->'0'])--weight('1')--> Transition 8
```

Obrázek 4.7: Příklad statistiky po krocích

Obrázek 4.8 pojednává o celkové souhrnné statistice. `Place count` značí celkový počet míst v zadaném modelu, `Transitio count` celkový počet přechodů v modelu a `Directeds count` celkový počet hran v modelu. `Simulation time` je čas běhu simulace.

```
********************** STATISTICS **********************
*       T
*       O    Places count : 12
*       T    Transitions count : 14
*       A    Directeds count : 27
*       L
*       S    Simulation time : 68.8996
```

Obrázek 4.8: Příklad souhrné statistiky

Na obrázku 4.9 je zobrazena statistika míst. Jednotlivé sloupce popořadě značí : pořadí prvků, jména prvků, kapacita míst, počáteční hodnota místa, koncová hodnota místa a poslední sloupec obsahuje celkový počet kuliček na daném místě.

```
* ------------------------------ places --------------------------
*
*      . |    name | capcity | start state | final state tags | total tags
*    ..................................................................
*    P 1 | Place 0 |      inf |           0 |                0 |     11
*    L 2 | Place 1 |      inf |           0 |                0 |     0
*    A 3 | Place 2 |      inf |           0 |                0 |     0
*    C 4 | Place 3 |      inf |           0 |                0 |     0
```

Obrázek 4.9: Příklad souhrnné statistiky Místa

Statistika přechodů je zobrazena na obrázku 4.10. Obsahuje všechny důležité vlastnosti přechodů : pořadí prvku, jména přechodů, prioritu, časovaný přechod, pravděpodobnost, počet značek vstupujících a vystupujících z přechodu.

Obrázek 4.11 představuje charakteristiku hrany s vlastnostmi jméno a váha hrany.

```
* ------------------------- transitions -------------------------
*
*    . |         name | priority | timed transition | probability | ...
*  ..................................................................
*  T 1 | Transition 0 |        0 |          exp(10) |           0
*  R 2 | Transition 1 |        0 |                5 |           0
*  A 3 | Transition 2 |        1 |                0 |           0
```

Obrázek 4.10: Příklad souhrnné statistiky Přechodů

```
* ---------------------- directed arc ----------------------
*
*    . |           name | weight
*  ..............................
*  D 1 | Directed arc 0 |      1
*  I 2 | Directed arc 1 |      1
*  R 3 | Directed arc 2 |      1
*  E 4 | Directed arc 3 |      1
```

Obrázek 4.11: Příklad souhrnné statistiky Hran

Kapitola 5

Shrnutí simulačních experimentů a závěr

Experimentováním s mnoha primitivními modely jsme dokázali správnost řešení simulace Petriho sítě s rozšířené o prioritu, pravděpodobnost a časování hran našim simulátorem. Výsledky simulací širokospektrálních testů pozůstávajících s modelů Petriho sítí zaměřených na odhalení případných chyb neodhalily žádnou, takže můžeme zkonstatovat, že daný simulátor se chová korektně a podává adekvátní výstupy zdokumentované v předešlých kapitolách. Některé validně zapsané Petriho sítě však není možné simulovat, proto tento simulátor obsahuje různé kontrolní mechanizmy na detekci takovýchto modelů a v konkretním případě upozorní na danou skutečnost uživatele, navrhne mu řešení a popřípadě informace o chybovém stavu.