PŘÍRODOVĚDECKÁ FAKULTA UNIVERZITY PALACKÉHO
KATEDRA INFORMATIKY

BAKALÁŘSKÁ PRÁCE

Stavové automaty

2012 Petr Jančík

Anotace

Tento text je doprovodný dokument ke grafickému editoru stavových automatů s výstupem. Obsahuje hlavně popis uživatelského rozhraní a vnitřního fungování programu. Program je napsán v jazyce C# v prostředí Microsoft Visual Studio 2008.

Chtěl bych poděkovat rodině za psychickou podporu a vedoucímu práce RNDr. Arnoštu Večerkovi za pomoc při vypracovávání.

Obsah

Seznam obrázků

Seznam tabulek

1. Zadání bakalářské práce

Cílem práce je sestavit aplikaci pro návrh stavových automatů (Mealyho automat, Mooreův automat). Návrh by měl být graficky (sestavením diagramu automatu) nebo formálním popisem automatu. Další funkcí aplikace by měla být simulace činnosti sestaveného automatu zobrazující jednotlivé kroky přijímání vstupního slova (přechody mezi stavy) a zobrazení výstupu automatu.

Můj přístup zahrnuje navrhnout grafický editor, který umožňuje návrh takovýchto strojů a simulaci průběhu jejich výpočtu. Jádro tvoří pracovní plocha, na které lze rychle a přehledně vytvořit stavy, přechody automatu a určit počáteční stavy výsledného automatu. Program obsahuje možnost zadat vstupní slovo a sledovat chování automatu na jeho přechodovém diagramu. Tento editor výrazně usnadní návrh strojů založených na stavových automatech a zvýší tak efektivitu práce.

2. Teoretický úvod

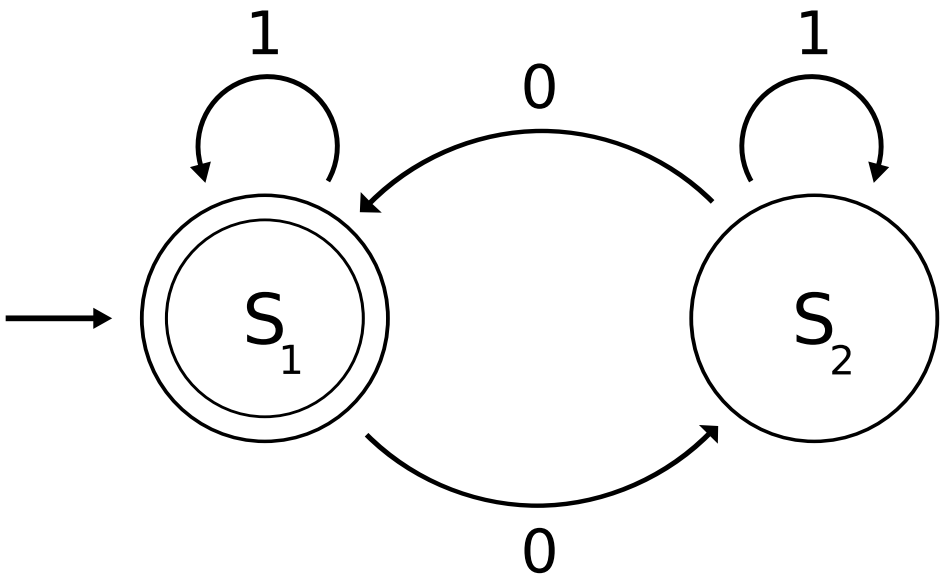
V této kapitole se podívám na typy stavových automatů, jejich příklady, popis činnosti a způsoby jejich zápisu.

2.1. Základy stavových automatů

Stavový automat, zkráceně KA (konečný automat) nebo FSM (finite state machine) je matematická abstrakce, používaná pro popis a návrh chování strojů, organismů nebo programů kolem nás. Je to zařízení, které přijímá vstupní informace a jako odpověď na ně dává informace výstupní. Přitom výstupy jednoznačně závisí pouze na vnitřním stavu automatu a vstupní informaci.[3] Teorie stavových automatů vychází z definice automatů o konečném počtu stavů, kterou položili Warren S. McCulloch a Walter S. Pitts v roce 1943. Na tuto práci navázali pánové George H. Mealy a Edward F. Moore v letech 1956 a 1955, po kterých jsou tyto automaty pojmenovány.

Stavové automaty se dělí na dvě hlavní skupiny. První skupinou jsou akceptory/přijímače. Tento typ automatu přijímá vstupní slovo, a pokud výpočet skončil/neskončil v přijímacím stavu, tak jako výsledek dává odpověď přijal/nepřijal. Tyto automaty jsou používány pro návrh parserů formálních jazyků, šifrovacích strojů, popis lidských jazyků a dalších. Příklad je na obrázku 1.

Druhá skupina jsou tzv. transducery, automaty, které přijímají vstup a na výstupu je sled akcí nebo výstupní slovo. V této skupině jsou automaty typu Mealy nebo Moore. Příkladem mohou být například stopky, automat na pití nebo složitější automaty, používané v průmyslové automatizaci.[3] Dále se budu zabývat pouze druhou skupinou.



Obrázek 1. Příklad akceptoru, přijímající sudý počet nul.

2.2. Formální definice

2.2.1. Stavový automat - akceptor

Akceptor je pětice (S, s_0, Σ, T, F) kde:

- S je neprázdná konečná množina stavů
- s_0 je počáteční stav z množiny S
- Σ je konečná vstupní abeceda
- T je přechodová funkce (T: $S \times \Sigma \to S$) další stav závisí na předchozím stavu a znaku na vstupu
- $F \subset S$ je množina přijímacích stavů

2.2.2. Mooreův stavový automat

Mooreův stavový automat s výstupem je šestice $(S, s_0, \Sigma, \Lambda, T, G)$ kde:

- S je neprázdná konečná množina stavů
- s_0 je počáteční stav z množiny S
- Σ je konečná vstupní abeceda
- Λ je konečná výstupní abeceda

- T je přechodová funkce ($T\colon S \times \Sigma \to S$) další stav závisí na předchozím stavu a znaku na vstupu
- G je výstupní funkce ($G\colon S \to \Lambda$) výstup závisí pouze na stavu, ve kterém se automat nachází

2.2.3. Mealyho stavový automat

Mealyho stavový automat s výstupem je šestice $(S, s_0, \Sigma, \Lambda, T, G)$ kde:

- S je neprázdná konečná množina stavů
- s_0 je počáteční stav z množiny S
- Σ je konečná vstupní abeceda
- Λ je konečná výstupní abeceda
- T je přechodová funkce ($T\colon S \times \Sigma \to S$)
- G je výstupní funkce ($G\colon S \times \Sigma \to \Lambda$) výstup závisí na stavu, ve kterém se automat nachází a na vstupním znaku

Formální definice převzaty z [2].

2.3. Chování stavových automatů

Chování automatů podle [3].

Automat se v každém okamžiku nachází v jednom stavu $q \in Q$. Stav automatu se mění po přečtení jednoho symbolu ze vstupu. Pokud je automat ve stavu $q \in Q$ a zpracuje vstupní znak $a \in \Sigma$, přejde do stavu $T(q, a)$. T tudíž nazýváme přechodovou funkcí. Mealyho automat vypíše výstupní znak $G(q, a)$ v okamžiku zpracování vstupního znaku a. Mooreův automat vydá výstup až po dosažení cílového stavu a ten je přístupný po celou dobu mezi příchody vstupních znaků. Práce automatu končí po zpracování celého vstupního řetězce.

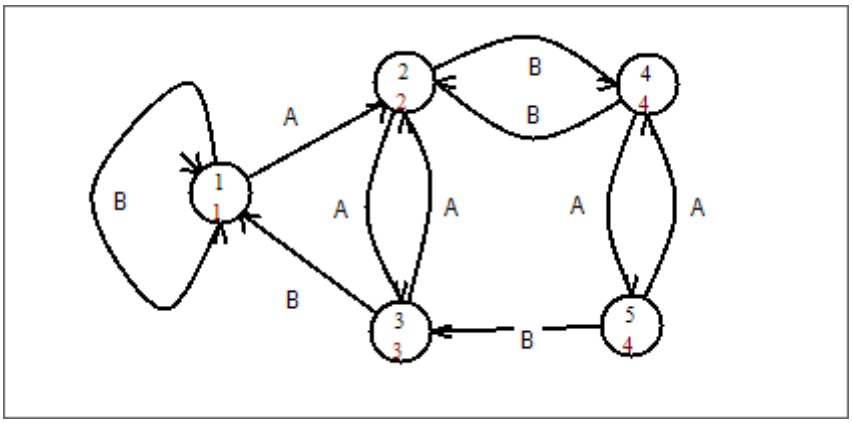
Jelikož lze o počítači přemýšlet jako o automatu, lze používat podobnou terminologii. Automat má konečnou vstupní pásku, na které je zapsáno vstupní slovo v. Páska má $n = l(v)$ políček, kde na každém je jeden znak vstupního slova. Po pásce se zleva doprava pohybuje čtecí hlava, která začíná na levém okraji pásky a s hodinovým taktem se pohybuje o jedno políčko. Automat má také výstupní pásku stejné délky, po které se pohybuje zleva doprava zapisovací hlava. Automat je ve stavu q. S hodinovým signálem čtecí hlava přečte písmeno a a automat přejde do stavu $T(q, a)$ a na výstupní pásku se vypíše u Mealyho automatu $G(q, a)$, u Mooreova znak z dosaženého stavu $G(T(q, a))$ a obě hlavy se posunou o políčko doprava. Avšak ne všechny automaty se musí s touto představou shodovat. V automatech používaných v reálných situacích mohou být vstupy a výstupy nahrazeny elektrickými signály od zařízení a hlavy mohou být tato zařízení.

2.4. Způsoby reprezentace přechodové funkce

Stavové automaty lze zadávat mnoha způsoby [2], které lze rozdělit do dvou skupin [3]:

- Zadání stavovou tabulkou:
  Tabulka Mealyho automatu má následující tvar: Řádky jsou nadepsány stavy, sloupce vstupy. V a-tém sloupci a q-tém řádku je dvojice $F(q,a)/G(q,a)$. Příklad tabulka 1.
  Tabulka Mooreova automatu má tvar: Řádky jsou stavy, sloupce vstupy a je přidán jeden sloupec pro výstup y V a-tém sloupci a q-tém řádku je $F(q,a)$ a v y je v q-tém řádku $G(q,a)$. Příklad tabulka 2.
  Počáteční stav q_0 je označen šipkou na jeho řádku.

- Zadání stavovým diagramem:
  Stavový diagram je ohodnocený multigraf, jehož vrcholy jsou stavy automatu.
  Pro vstup a a vrchol q Mealyho automatu vede ze stavu q do stavu $F(q,a)$ hrana ohodnocená dvojicí $a/G(q,a)$. Příklad obrázek 3.
  Pro vstup a a vrchol q Mooreova automatu vede ze stavu q do stavu $F(q,a)$ hrana ohodnocená vstupem a. Dále je každý vrchol ohodnocen $G(q,a)$. Příklad obrázek 2.



Obrázek 2. Stavový diagram Mooreova automatu - Stopky

2.5. Převody automatů

Jelikož jsou oba typy automatů ekvivalentní, viz [3], lze je mezi sebou převádět [4].

	Vstupy		
Stavy	a	b	c
10	$5/s$	$8/v$	$9/y$
9	$4/s$	$7/v$	$8/y$
8	$3/s$	$6/v$	$7/y$
7	$2/s$	$5/v$	$6/y$
6	$1/s$	$4/v$	$5/y$
5	$0/s$	$3/v$	$4/y$
4	$4/t$	$2/v$	$3/y$
3	$3/t$	$1/v$	$2/y$
2	$2/t$	$0/v$	$1/y$
1	$1/t$	$1/w$	$0/y$
0	$0/t$	$0/w$	$0/z$

Tabulka 1. Stavová tabulka Mealyho automatu - Automat na jízdenky

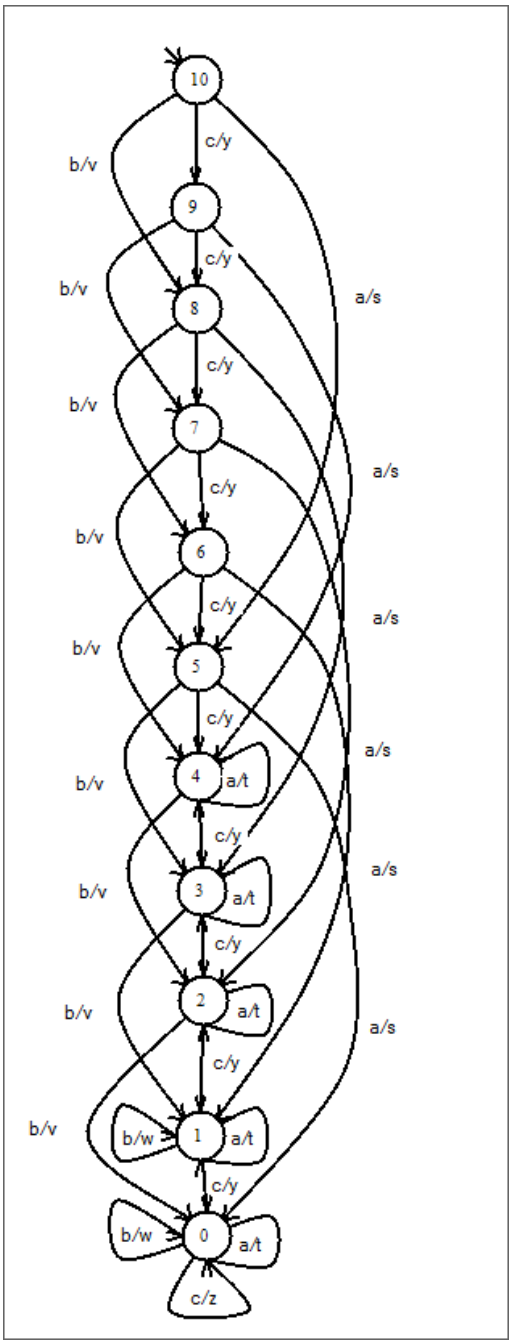
	Vstupy		**Výstup**
Stavy	**A**	**B**	**y**
1	2	1	1
2	3	4	2
3	2	1	3
4	5	2	4’
5	4	3	4’

Tabulka 2. Stavová tabulka Mooreova automatu - Stopky

2.5.1. Moore $->$ Mealy

Mooreho automat převedeme na Mealyho tak, že hodnoty výstupů uložené u stavů přepíšeme k hranám, které do těchto uzlů směřují. V takto upraveném automatu jsou hodnoty výstupů v každém stavu stále plně určeny tímto stavem. Takovýto automat je možné zjednodušit sloučením hran a stavů, které to dovolují.

Lze sloučit hrany, které vycházejí i končí ve stejných stavech a mají stejné hodnoty výstupů. Přitom podmínka (vstup) přechodu u sloučené hrany bude rovna logickému součtu podmínek slučovaných hran (při obou vstupech půjdeme toutéž hranou). Stavy lze sloučit, pokud hrany, které z nich vycházejí, vstupují do stejných dalších stavů a mají přiřazeny stejné výstupní hodnoty. Touto úpravou vznikne Mealyho automat.



Obrázek 3. Stavový diagram Mealyho Automatu - Automat na jízdenky

2.5.2. Mealy $->$ Moore

Převod z Mealyho na Mooreův automat provedeme tak, že rozštěpíme stavy automatu, do kterých směřují hrany s odlišnými výstupními hodnotami. Do každého nového stavu budou vcházet pouze hrany se stejným výstupem. Poté pouze

přesuneme výstupní hodnoty do stavů a máme Mooreův automat. Tento převod se v praxi obvykle neprovádí, protože Mealyho automat je elektronicky přívětivější, a jde tedy spíše o teoretickou možnost.[4]

3. Uživatelská část

3.1. Požadavky na spuštění

Ke spuštění aplikace je potřeba Windows XP, Vista nebo Windows 7 s nainstalovanou .NET Framework 3.

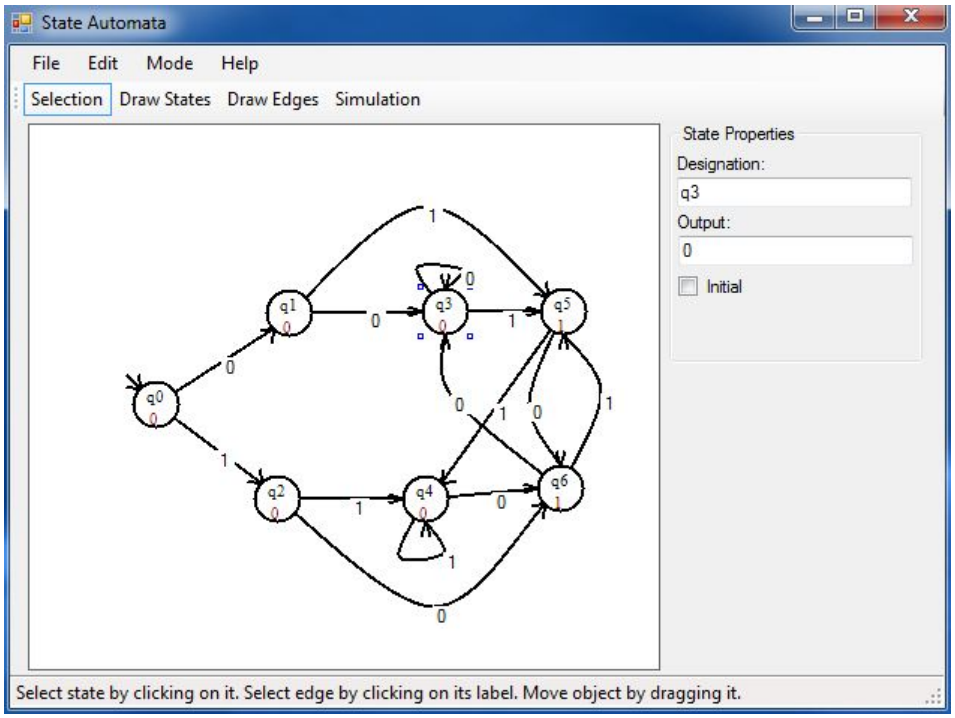
Windows 7 obsahuje .NET Framework 3.5 a Windows Vista obsahuje .NET Framework 3, tudíž jen uživatelé Windows XP si ji musí nainstalovat z přiloženého DVD nebo stáhnout nejnovější ze stránek Microsoftu.

Aplikace se spouští souborem `StateAutomata.exe`. Pro nápovědu je potřeba mít v umístění souboru také soubor s nápovědou `StateAutomata.chm`.

3.2. Popis programu

Tento program slouží k designu stavových automatů s výstupem. Umožňuje pohodlné a přehledné sestavení Mooreova nebo Mealyho stavového automatu a definici vstupních a výstupních znaků. Designované automaty je možné kdykoli načíst nebo uložit.

3.3. Ovládání



Obrázek 4. Uživatelské rozhraní, Mooreův automat

3.3.1. Popis okna

- *Menu*
  V horní části okna je hlavní menu aplikace. Viz obrázek 4.
  - V položce File jsou možnosti vytvoření nového Mooreova nebo Mealyho automatu, možnost načíst automat ze složky disku. Dále umožňuje uložit automat do souboru nebo jako obrázek viditelné časti. Poslední položka slouží k uzavření aplikace.
  - V položce Edit je možno smazat označený stav/hranu v hlavním okně.
  - Položka menu Mode obsahuje možnosti módů aplikace.
  - Poslední položkou je Help, která obsahuje odkaz na tento dokument.
- *Nabídka módů*
  Pod menu se nachází snadno přístupná lišta s módy aplikace.
- *Panel nastavení objektu / ovládání simulace*
  - Pokud je označen objekt, tato nabídka zobrazuje a umožňuje měnit jeho vlastnosti.
  - Při označení stavu se zde objeví jeho název, možnost zvolit jej jako počáteční a pokud je právě upravován Mooreův automat, tak i výstupní znak.
  - Při označení hrany se zobrazí její vstupní znak a pokud je automat Mealyho, tak také její výstup.
  - Do polí vstupu a výstupu je možné zadat libovolné znaky, pokud je zadán více než jeden znak, tak jsou všechny znaky mimo první ignorovány.
  - V módu simulace je tato oblast využita pro její ovládání. Je možné zadat vstupní slovo, po přečtení celého vstupu pak lze kopírovat výstupní slovo. Objeví se také 4 tlačítka pro posun na počátek simulace, krok zpět, krok vpřed a posun na konec simulace.
- *Pracovní plocha*
  Pracovní plocha zabírá v rozhraní největši prostor a je hlavním prvkem, se kterým uživatel pracuje. Podle módu aplikace umožňuje selekci objektů, přidávání stavů, hran a sledování průběhu simulace.

3.3.2. Módy aplikace

- Selection - umožňuje vybírat objekty.
  Vybrané objekty lze posouvat tažením myší a měnit jejich vlastnosti. Stav se označuje kliknutím na něj, hrana pak kliknutím na její popisku. Odznačení se provádí klikem pravého tlačítka do pracovní plochy.

- DrawStates - umožňuje přidávat stavy k automatu.
  Stav se přidává kliknutím na volné místo na pracovní ploše. Je možné ihned měnit vlasnosti přidaného stavu. Při kliknutí na existující stav se daný stav označí.

- DrawEdges - umožňuje vytvářet přechodové hrany mezi stavy automatu. Kliknutím na existující stav se začne vytvářet hrana z tohoto stavu. Při vytváření hrany lze přidávat další body kliknutím na volnou plochu. Přidání stavu je dokončeno opětovným klikem na některý ze stavů. Kliknutím pravým tlačítkem do plochy lze zrušit právě zakreslovanou hranu.

- Simulation - v tomto módu lze simulovat výpočet automatu.

3.3.3. Klávesové zkratky

- Delete - Smaže označený objekt.

- F1 - Zobrazí nápovědu.

- Ctrl+1 - Přepne do módu Selection.

- Ctrl+2 - Přepne do módu DrawStates.

- Ctrl+3 - Přepne do módu DrawEdges.

- Ctrl+4 - Přepne do módu Simulation.

- Ctrl+S - Uloží automat do souboru.

- Ctrl+O - Otevře automat ze souboru.

3.3.4. Průvodce

Při spuštění aplikace se automaticky vytvoří nový Mooreův automat. Pokud požadujeme jiný typ automatu, lze jej vybrat v menu File nebo můžeme načíst již hotový automat z disku.

Program začíná v módu `Selection`. Klikem na lištu módů nebo v menu Mode si vybereme mód kreslení stavů (`Draw States`) a na ploše vytvoříme stavy, které by měl námi požadovaný automat mít. U každého stavu nastavíme jeho označení a u Mooreova stroje také výstupní znak stavu. Stav je možno označit klinutím na něj a posouvat tažením na volnou pozici.

Poté se přepneme do módu kreslení hran (`Draw Edges`), a mezi stavy vytvoříme hrany. Lze vytvářet přímé hrany nebo při vytváření hrany pomocí klikání na

prázdnou plochu přidat další body a vytvořit tak zaoblené hrany. U hran nastavíme vstup, při kterém bude hranou procházet výpočet. U Mealyho stroje navíc přidáme výstupní znak při průchodu hranou. Hrany je možné označit kliknutím na jejich popisku. Pokud není hrana přímá, pak se při označení objeví body, kterými lze tvarovat výslednou křivku.

Nyní, pokud jsme s automatem spokojeni, můžeme přejít do stavu simulace, jinak doplníme další stavy a hrany. V režimu simulace zadáme vstupní slovo a tlačítkem `>` provedeme první krok výpočtu. Pokud nechceme krokovat výpočet a rádi bychom zjistili, jaký bude sled výstupů po skončení výpočtu, klikneme na tlačítko `>>` které nás přesune na konec simulace. Pokud je v automatu chyba, například chybí hrana pro daný vstup, program při pokusu o přechod do módu `Simulation` nahlásí chybu, kterou je třeba opravit. Po ukončení výpočtu je možné si výstupní řetězec zkopírovat. Pokud chceme vyzkoušet jiný vstup, lze se tlačítkem `<<` vrátit na začátek a změnit vstupní slovo.

Jakmile jsme spokojeni s funkčností stroje, lze jej uložit v menu File na požadované místo na disk. Automat lze také uložit ve formě obrázku. POZOR! Na výsledném obrázku bude pouze viditelná část automatu.

4. Programátorská část

Program je napsán v jazyce C# za použití standardních knihoven .NET.

4.1. Popis programu a enumerací

Aplikace se skládá ze tříd pro reprezentaci stavů (`State`) a hran (`Edge`), které dědí ze společné třídy prvku automatu (`GraphElement`). Automat sám je uložen ve třídě `Machine`. Simulace automatu je obstarána třídou `Simulator`. Třída uživatelského rozhraní `Form1`, která dědí z obecného formuláře, obstarává ovládání aplikace, jako je vykreslování automatu na plochu, obsluhu klikání, změnu parametrů a nápovědu.

Program obsahuje tři enumerace:

- `MachineType`, která má hodnoty pro oba typy automatů, `Mealy` a `Moore`.
- `Mode`, je enumerace módů a obsahuje hodnoty `Selection, DrawStates, DrawEdges, Simulation` pro jednotlivé módy.
- `ArrowArientation`, udává orientaci šipky hran.

4.2. Popis tříd

4.2.1. GraphElement

Třída prvku grafu obsahuje data společná pro stavy a hrany, informace o výstupu prvku a o pozici prvku na pracovní ploše.

Virtuální metoda `BelongsTo(point)` zjišťuje, zda se bod, na který bylo kliknuto, nachází uvnitř tohoto prvku. Druhá virtuální metoda `Move(X, Y)` pohybuje prvkem na místo dané souřadnicemi X a Y. Implementace těchto metod jsou u stavu a hrany odlišné, a proto jsou tyto metody v potomcích přepsány.

4.2.2. State

Ve třídě stavu jsou informace specifické pro stav. Stavy jsou reprezentovány na pracovní ploše kružnicemi, které mají střed v bodě `Center` a poloměr `Radius`. Každý stav má uvnitř své označení `Designation` a u Mooreova automatu pak i výstup. Dále obsahuje seznamy hran které vstupují a vystupují z daného stavu. Tyto seznamy jsou používány při posouvání stavu po ploše (pro jejich překreslení) a při simulaci (pro přechod mezi stavy). Po přesunu jednoho ze stavů hrany je navíc třeba upravit pozici prvního a posledního bodu ve hranách vstupujících a vystupujících z posunutého stavu a zjistit, zda není potřeba upravit orientaci šipky do něj směřující.

4.2.3. Edge

Hrana s sebou nese reference na její počáteční `From` a koncový `To` stav. Dále obsahuje seznam bodů `Path` pro vykreslení hrany na pracovní plochu, vstupní znak `Input` pro přechodovou funkci automatu a popisku `Label`, která vstupní a u Mealyho automatu i výstupní znak vykresluje na plochu.

Při posunu bodu hrany se index tohoto bodu v `Path` uloží v objektu popisky a metoda `Move(X, Y)` tuto informaci použije k nahrazení předchozího bodu bodem posunutým. Metoda `UpdateArrow(RemoveLastPoint)` při vytváření hrany nebo při jejím posunu nastavuje pozici posledního bodu a orientaci šipky směřující do koncového stavu hrany.

4.2.4. Machine

Objekty třídy `Machine` obsahují seznam stavů `States` a seznam hran `Edges`. Typ automatu `MType` určuje chování automatu, jeho zobrazeni a zda je výstup automatu u hrany nebo ve stavu. Počáteční stav je uložen ve slotu `InitialState`.

Metoda `ValidMachine()` ověřuje, zda je automat kompletní. Automat je kompletní pokud má alespoň jednu hranu a z každého stavu vychází hrany se všemi vstupy (determinismus). Použité symboly získá pomocná metoda `AccumulateInputs()`. Před započetím simulace je také potřeba zkontrolovat pomocí `ValidInputString(InputString)`, jestli vstupní řetězec neobsahuje neplatné symboly.

4.2.5. Simulator

Třída `Simulator` obsahuje vše, co je potřeba k simulaci činnosti automatu; slot na stav, ve kterém je výpočet `CurrentState`, řetězec na vstupu automatu `Input`, výstupní řetězec po skončení výpočtu `Output` a flag `SimulationInPoregress`, který říká ostatním objektům, že simulace už běží.

4.2.6. Form1

Tato třída obsahuje uživatelské rozhraní. `ProgramMode` označuje, ve kterém ze čtyř módu se program nachází. Konstanty umožňují upravovat parametry vykreslování automatu do pracovní plochy. Uživatelské rozhraní obsahuje také reference na objekty automatu `Machine`, na kterém se pracuje, a na simulátor `Simulator`, který zprostředkovává simulaci automatu. Slot `SelectedObject` obsahuje právě označený prvek grafu, který je na pracovní poše kreslen modrou barvou a jsou kolem něj nakresleny modré čtverce pro odlišení od neoznačených prvků.

4.3. Implementační detaily

4.3.1. Posouvání objektů

Při stisknutí levého tlačítka do pracovní plochy se nejdříve označí objekt pod kurzorem. Pokud bylo kliknuto na stav a není rozpracovaná cesta, tak se nastaví flag posouvání. Pokud je zvolena hrana, tak se testuje, zda nebyl při stisku tlačítka kurzor v posouvacím rámečku u bodu hrany. Pokud byl, tak se uloží index bodu do objektu popisky hrany a taktéž se nastaví flag posouvání. Při povolení tlačítka se otestuje flag posouvání a také se otestuje, zda se vůbec mezi stiskem a povolením kurzor pohnul. Pokud ano, tak se daný objekt posune, a panel se překreslí.

4.3.2. Kliknutí do pracovní plochy

Při kliknutí pravým tlačitkem se zruši veškeré akce (tvorba hrany, tažení objektem) a odznačí objekty.

Při kliknutí levým tlačítkem se podle módu programu stane následující:

- `Selection` - pouze se označí objekt pod kurzorem.
- `DrawStates` - pokud na místě kliknutí není stav, tak se vytvoří nový stav, přidá se do automatu a označí se.
- `DrawEdges` - Viz Tvorba hrany.

4.3.3. Tvorba hrany

Pokud zatím netvoříme hranu a je kliknuto na stav, tak se započne hrana z tohoto stavu a nastaví se flag tvorby hrany.

Pokud již tvoříme hranu, tak každým kliknutím na volné místo přidáme bod hraně. Hrana se během přidávání bodů z důvodu náročnosti překreslování a blikání nepřekresluje.

Hranu ukončíme kliknutím na některý ze stavů, čímž se resetuje flag tvorby hrany. Při dokončování hrany zjistíme, jestli tvořená hrana nepřekruje nekterou z již existujících přímých hran(hran se dvěma body), a popřípadě zahlásíme chybu.

Dokončením hrany se vytvoří popiska hrany, nastaví se vstupní znak hrany, hrana se ukončí šipkou a vykreslí se na plochu.

Při tvorbě hrany, která začíná i končí ve stejném stavu, stačí buď dvakrát kliknout na stejný stav, nebo jedním mezibodem určit její směr.

4.3.4. Označení objektů

Při kliknutí na stav nebo popisku hrany se daný objekt uloží do slotu `SelectedObject` a obarví se barvou, obsaženou v `selectionPen`. V panelu napravo od plochy se vyplní informace o objektu a umožní tak upravit u stavu označení a výstup, a u hrany vstup a výstup.

4.3.5. Překreslování pracovní plochy

Při práci se musí pracovní plocha průběžně překreslovat. Například posunem stavu musíme překreslit stav samotný a jeho hrany. Překreslování probíhá podle potřeby v celém panelu, nebo jen jeho změněné části pomocí metody panelu `Invalidate(Region)` která vyvolá událost překreslení obslouženou metodou `panel1_Paint(sender, e)`. První se překreslí všechny hrany a poté všechny stavy. U počátečního stavu se nakreslí šipka. Označená hrana se vždy překreslí modrou barvou, u stavu se nakreslí v rozích modré čtverce. V režimu simulace se navíc stav, ve kterém je výpočet, vybarví šedou barvou.

4.3.6. Simulace automatu

Základní ovládání simulátoru obstarávají metody simulátoru `Forward()` a `Back()`, které umožňují krokovat výpočet. Metoda `Forward()` přečte znak ze vstupu a najde hranu, kterou se má výpočet ubírat dále, uloží nynější stav do historie, vypíše na výstup příslušný znak, označí koncový stav hrany za `CurrentState` a otestuje pomocí `CanGoForward()`, jestli ještě není na konci simulace. Pokud už je výpočet na konci, tak se v uživatelském rozhraní deaktivují tlačítka pro pokračování ve výpočtu. Metoda `Back()` se vrací ve výpočtu o jeden krok tak, že poslední stav z historie odebere a označí jako `CurrentState`, z výstupu odebere jeden znak, na vstup vrátí zpracovaný znak. Pak pokud už je na začátku simulace (`CanGoBack()` je false), tak znemožní jít zpět.

Pokročilejší ovládací metody `SimulateToEnd()` a `StartOver()` umožňují rychle ukázat výsledek simulace a vrátit se na začátek simulace, kde je možné znova zadat vstupní řetězec.

4.3.7. Uložení a načtení automatu

Program umožňuje uložit a načíst automat ve formě XML dokumentu. Automat lze také uložit ve formě obrázku. Pro správné uložení do obrázku je třeba mít v pracovní ploše celý automat, jinak bude na obrázku pouze viditelná část automatu. Důvodem je použití vykreslovací funkce panelu `DrawToBitmap`.

Závěr

Výsledkem této práce je naprogramovaný editor stavových automatů s výstupem. Editor obsahuje všechny požadované funkce v grafickém uživatelském rozhraní, zpracovaném podle standardů Windows User Interface Guidelines.

Programová stránka aplikace je ve funkčním stavu, ale několik věcí by bylo možné vylepšit. Bylo by vhodné přidat vykreslování tažených objektů během tažení na novou pozici a vykreslování neuplné hrany při její tvorbě. V nynější implementaci by tyto úpravy způsobily blikání.

Program by také mohl být rozšířen přidáním lokalizovaných menu a možností komplexních Mooreových a Mealyových strojů s více vstupy a výstupy.

Conclusions

The result of this bachelor thesis is an editor of state automata with output, called finite-state transducers. It includes all the required functions in a graphical user interface, made acording to Windows User Interface Guidelines.

The programing side of the application is in working order, but things could be improved. It would be appropriate to redraw objects while they are dragged to their new position and to render incomplete edges during creation. These changes would cause image flickering in the current implementation.

The program could also be extended by adding localized menus and a possibility to create complex Mealy and Moore machines with multiple inputs and outputs.

Reference

[1] Pilgrim, Robert A. *Computing Machinery* Robot Crafters, LLC, 2006.

[2] *Wikipedia*, články *Moore machine*, *Mealy machine* a *Finite-State machine*.

[3] Demlová, Marie; Koubek, Václav *Algebraická teorie automatů* SNTL, Praha, 1990.

[4] Přednáška *Transformace stavových automatů* na VUT Brno.

[5] Ľudovít, Molnár; Češka, Milan; Melichar, Bořivoj *Gramatiky a jazyky* SNTL, Praha, 1987.

A. Obsah přiloženého CD

`bin/`
: Obsahuje program StateAutomata ve spustitelné podobě přímo z CD/DVD. Adresář obsahuje i všechny potřebné knihovny a další soubory pro bezproblémové spuštění programu.

`doc/`
: Dokumentace práce ve formátu PDF, vytvořená dle závazného stylu KI PřF pro diplomové práce, včetně všech příloh, a všechny soubory nutné pro bezproblémové vygenerování PDF souboru dokumentace (v ZIP archivu), tj. zdrojový text dokumentace, vložené obrázky, apod.

`src/`
: Kompletní zdrojové texty programu se všemi potřebnými (převzatými) zdrojovými texty, knihovnami a dalšími soubory pro bezproblémové vytvoření spustitelných verzí programu.

`readme.txt`
: Instrukce pro instalaci a spuštění programu, včetně požadavků pro jeho provoz.

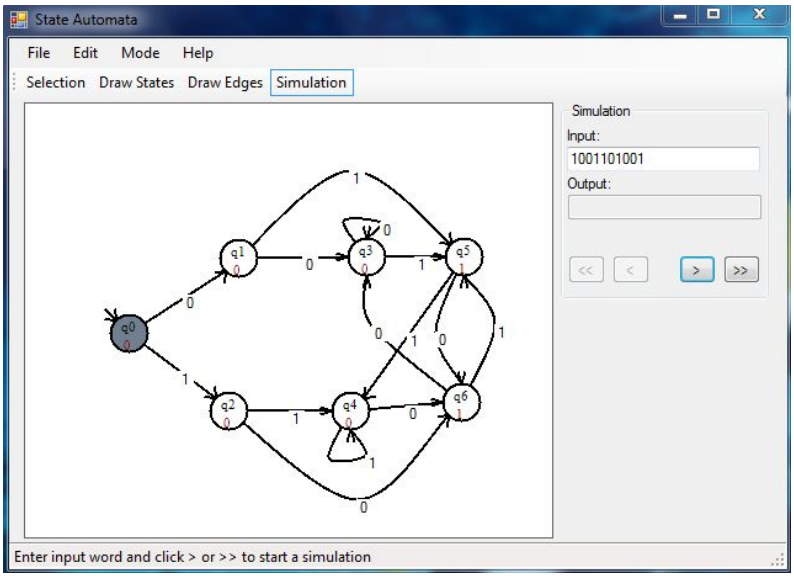
Navíc CD/DVD obsahuje:

`data/`
: Ukázková a testovací data použitá v práci a pro potřeby obhajoby práce.

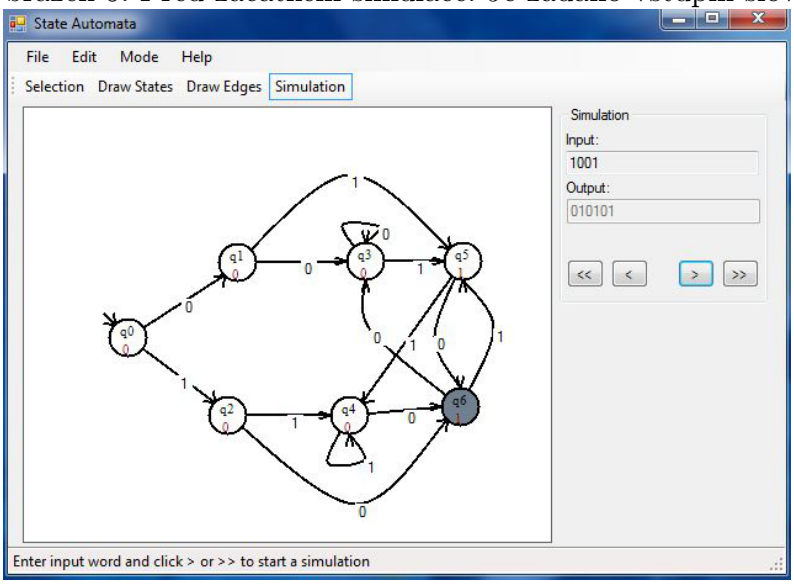
`install/`
: Instalátory aplikací, knihoven a jiných souborů nutných pro provoz programu / webové aplikace, které nejsou standardní součástí operačního systému.

U veškerých odjinud převzatých materiálů obsažených na CD/DVD jejich zahrnutí dovolují podmínky pro jejich šíření nebo přiložený souhlas držitele copyrightu. Pro materiály, u kterých toto není splněno, je uveden jejich zdroj (webová adresa) v textu dokumentace práce nebo v souboru `readme.txt`.

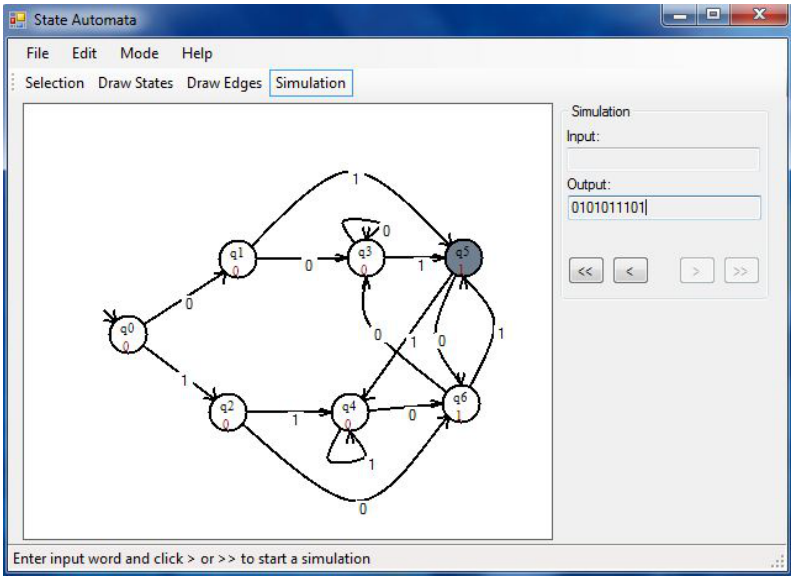
B. Ukázky simulace



Obrázek 5. Před začátkem simulace. Je zadáno vstupní slovo



Obrázek 6. V průběhu simulace jsou políčka pro vstup a výstup nedostupné.



Obrázek 7. Na konci simulace je pole výstupu otevřeno pro čtení.