České vysoké učení technické v Praze
Fakulta jaderná a fyzikálně inženýrská

Výzkumný úkol

Teorie a aplikace multiagentního řízení dopravy

Vypracoval: Jakub Novotný
Vedoucí práce: Dr. Ing. Jan Přikryl, Ph.D.
Konzultant: Ing. Václav Šmídl, Ph.D.

Praha, 2011

Prohlašuji, že jsem předloženou práci vypracoval samostatně, a že jsem uvedl veškerou použitou literaturu.

Praha, 8. července 2010 Jakub Novotný

Rád bych poděkoval vedoucímu práce Ing. Janu Přikrylovi, Ph.D. a odbornému konzultantovi Ing. Václavu Šmídlovi, Ph.D. za odborné vedení a úsilí vynaložené při tvorbě této práce.

Název práce:
Teorie a aplikace multiagentního řízení dopravy

Autor: Jakub Novotný

Obor: Inženýrská informatika
Zaměření: Tvorba softwaru
Druh práce: Výzkumný úkol

Vedoucí práce: Dr. Ing. Jan Přikryl, Ph.D.

Abstrakt:
V této práci je ...

Klíčová slova:
multiagentní systémy, decentralizované řízení, řízení dopravy

Title:
Theory of multiagent traffic control

Author: Jakub Novotný

Abstract:
This work represents ...

Key words:
multiagent systems, decentralized control, control of traffic lights

Zadání práce s podpisem děkana

Vypracujte přehled literatury zabývající se použitím multi-agentních systémů v řízení dopravy. Zvláštní pozornost věnujte řízení v městském provozu s množstvím křižovatek. Z použitých přístupů sestavte přehled používaných metod autonomního řízení a komunikovaných informací mezi agenty. Z dostupných metod zvolte takovou, která je v vhodná pro použití v oblasti Zličína a implementujte ji v prostředí řízení mikrosimulátoru AIMSUN. Výsledné řízení zvolenou metodou srovnejte se současným řídícím algoritmem navrženým dopravními experty.

Obsah

Seznam obrázků

Seznam tabulek

Kapitola 1

Úvod

Multiagentní systémy jsou dnes rychle se rozvijející formou decentralizované umělé inteligence a s úspěchem se aplikují na celou řadu problémů, kde není možné použít centrální řízení a je zapotřebí komunikace, koordinace a spolupráce.

V této práci si nejdříve definujeme základní pojmy, a poté se hlouběji podíváme na problém interakce agentů. Popíšeme zde různé způsoby výběru vhodných strategií za pomoci komunikace, teorie her a modelování chování. Prozkoumáme některé způsoby učení agentů, které pomáhají zlepšovat jejich reakce na neustále se měníci prostředí a na chování ostatních.

V další části prozkoumáme nasazení těchto metod k řízení dopravy a zhodnotíme úspěch těchto aplikací.

Kapitola 2

Multiagentní systémy

2.1 Úvod

Multiagentní systém je druh distribuované umělé inteligence. Tento systém se skládá z jednotlivých výpočetních prvků, tzv. agentů, které musí mít dvě základní schopnosti. Zaprvé musí být schopni autonomní akce rozhodnutí - zjistit jak nejlépe dosáhnout požadovaných cílů - a zadruhé je to schopnost interakce s ostatními agenty. V druhém případě nejde jen o pouhou výměnu dat, ale o typ kolektivní aktivity - návrh, potvrzení, odmítnutí.

2.1.1 Historie

Multiagentní systémy jsou na poli počítačové vědy relativní novinkou. Studium tohoto tématu probíhá od začátku osmdesátých let dvacátého století. Větší pozornosti se jim dostalo v polovině let devadesátých s rozvojem internetu.

2.1.2 Agent

Neexistuje obecně uznávaná definice agenta. Přikloníme se k definici použité v publikaci [14].

Definice 2.1 (Agent): Agent je počítačový systém umístěný do nějakého prostředí, který je schopen autonomní akce k přiblížení se navrženým cílům. ▶

Agent v naprosté většině případů nemá celkovou kontrolu nad prostředím. Prostředí může ovlivňovat jen částečně. Obecně je prostředí nedeterministické. Stejná akce provedená dvakrát za sebou nemusí vést ke stejnému výsledku.

2.2 Druhy prostředí

Způsob práce agentů se liší podle druhu prostředí, ve kterém pracují. Podle [14] se prostředí dají klasifikovat následovně:

- Deterministické vs. nedeterministické
- Dostupné vs. nedostupné
- Statické vs. dynamické

Deterministické prostředí je takové, ve kterém má každá jednotlivá akce předem daný efekt. Prostředí je dostupné, pokud agent může zjistit jeho úplný stav v kteroukoliv dobu. Statické prostředí se na rozdíl od dynamického mění pouze vlivem akcí vyvolanými agenty. V diskrétním prostředí existuje pevné konečné číslo možných vjemů a akcí.

2.3 Interakce agentů

2.3.1 Stavy prostředí a preference agentů

Mějme pro jednoduchost 2 agenty. Označme si je i a j. Předpokládejme, že máme množinu

$$\Omega = \{\omega_1, \omega_2, ...\}$$

obsahující všechny možné stavy prostředí, v kterém agenti operují. Aby byl agent schopen efektivně ovlivňovat prostření, musí být schopen ohodnotit, jak je pro něj daný stav příznivý. Hodnocení daného stavu agenta i a j formálně definujeme jako funkce

$$u_i : \Omega \to \mathbb{R},$$

$$u_j : \Omega \to \mathbb{R}.$$

Čím je stav ω příznivější pro agenta i, tím je větší hodnota funkce u_i.

Definice 2.2 (Uspořádání na množině všech stavů): Mějme 2 stavy prostředí ω_1, ω_2. Řekněme, že stav ω_1 je preferován agentem i nad stavem ω_2, pokud platí $u_i(\omega_1) \geqq u_i(\omega_2)$. Značíme

$$\omega_1 \succeq_i \omega_2.$$

Stav ω_1 je silně preferován agentem i nad stavem ω_2, pokud platí $u_i(\omega_1) > u_i(\omega_2)$. Značíme

$$\omega_1 \succ_i \omega_2$$

Relace $\succeq_i$ je zřejmě uspořádání, protože má všechny potřebné vlastnosti.
Reflexivitu:

$$\forall \omega \in \Omega : \omega \succeq_i \omega$$

Tranzitivitu:

$$\forall \omega_1, \omega_2, \omega_3 \in \Omega : \omega_1 \succeq_i \omega_2 \wedge \omega_2 \succeq_i \omega_3 \Rightarrow \omega_1 \succeq_i \omega_3$$

Porovnatelnost:

$$\forall \omega_1, \omega_2 \in \Omega : \omega_1 \succeq_i \omega_2 \vee \omega_2 \succeq_i \omega_1$$

Relace $\succ_i$ zjevně nesplňuje podmínky reflexivity.

Kapitola 3

Výběr strategie genta

Nyní popíšeme, jak mají agenti možnost ovlivňovat prostředí. Opět předpokládejme existenci dvou agentů i a j. Obecně mají různí agenti různou oblast působnosti. Množina

$$A = \{a_1, a_2, ...\}$$

znázorňuje množinu všech akcí, které jsou agenti schopni provézt. Na tyto akce reaguje prostředí přechodem do nějakého stavu $\omega \in \Omega$. Formálně můžeme tento přechod zapsat jako funkci

$$\tau : A \times A \to \Omega.$$

3.1 Výběr strategie podle teorie her

Popišme zde způsob, jak se agent rozhodne pro realizaci určité akce. Agent je nyní schopen ohodnotit, který stav prostředí je pro něj příznivější než jiný, neví však jak budou reagovat ostatní agenti, není schopen určit tudíž, i za předpokladu, že by systém byl deterministický, do jakého stavu systém přejde. K výběru optimální akce se používají prvky z teorie her. Zadefinujme nejprve v souladu s touto teorií základní pojmy.

Definice 3.1 (Dominance množiny): Mějme 2 podmnožiny $\Omega_1, \Omega_2 \subset \Omega$. Řekneme, že Ω_1 je pro agenta i dominantní nad množinou Ω_2, pokud platí

$$\forall \omega \in \Omega_1, \forall \omega' \in \Omega_2 : \omega \succeq_i \omega'.$$

Řekneme že Ω_1 je pro agenta i silně dominantní nad množinou Ω_2, pokud platí

$$\forall \omega \in \Omega_1, \forall \omega' \in \Omega_2 : \omega \succ_i \omega'.$$

Abychom používali terminologii teorie her, budeme nyní akce $a_i \in A$ jednotlivých agentů nazývat strategiemi.

Definice 3.2 (Množina výsledků): Nazvěme množinu všech stavů, do kterých může prostředí přejít při hraní strategie $a_i \in A$, množinou možných výsledků. Označme ji

$$a_i^* \subset \Omega.$$

Definice 3.3 (Dominance strategie): Řekneme, že strategie a_i je dominantní nad strategií a_j, pokud je množina a_i^* dominantní nad množinou a_j^*. Strategie a_i je silně dominantní nad strategií a_j, pokud je množina a_i^* silně dominantní nad množinou a_j^*. ▶

Racionálně uvažující agent tedy vyloučí všechny strategie a_i, jestliže existuje strategie a_j, která nad strategií a_i silně dominuje. K zúžení výběru zbývajících strategií slouží Nashova rovnost. Pro zjednodušení uvažujme 2 agenty, i a j

Definice 3.4 (Nashova rovnost): Dvě strategie, a_1 a a_2 jsou v Nashově rovnosti, pokud za předpokladu že agent i zvolí strategii a_1, je nejvýhodnější strategií pro agenta j je strategie a_2 a zároveň pokud agent j zvolí strategii a_2, je pro agenta i nejvýhodnější strategií a_1. ▶

3.2 RMM - Rekurzivní modelové metody

Rekurzivní modelová metoda, která byla použita v článku [4] k modelování chování agentů ovládajících ostatní dopravní uzly, slouží k odhadu chování ostatních a gentů. Akce každého agenta z pravidlo ovlivňuje do určité míry celý systém, tudíž výběr strategie každého agenta závisí na předpokládaném chování ostatních agentů. Tato metoda minimalizuje nutnost komunikace a vyjednávání o provedení jisté akce tím, že každý agent je schopen modelovat rozhodnutí ostatních a podle známých parametrů prostředí s určitou pravděpodobností stanovit jejich volbu.

3.2.1 Formální definice

Základním stavebním kamenem RMM je matice zisků agenta, Definovaná v souladu s teorií her v [4]

Definice 3.5 (Matice zisků): Matice zisků P_{R_i} agenta R_i je definována trojicí

$$(R, A, U)$$

kde R je množina všech agentů v systému, A je množina množin $A_j = \{a_1^j, a_2^j, ...\}$ alternativních akcí agenta R_j. A_j budeme nazývat rozhodovací prostor agenta R_j. U je funkce

$$U : A_1 \times A_2 \times ... \times A_n \to \mathbb{R}$$

přiřazující hodnoty zisku všem kombinacím akcí všech agentů. ▶

Každý agent provádí danou akci z nějakého důvodu. Zisky agenta R_i jsou spojeny s provedením jeho určité akce $a_m^i \in A_i$ za předpokladu, že ostatní agenti $R_j, j \in \{1..n\}/\{i\}$ provedou akci $a_k^j \in A_j$. Matice je tedy n-dimenzionální, kde n je počet agentů v systému, a sestává se z prvků $u^{R_i}_{a_k^1...a_l^i...a_m^n}$, reprezentující zisky v dané situaci.
K určení pravděpodobnosti provedení strategií ostatních agentů se definuje rekursivní modelová struktura. [4]

Definice 3.6 (Rekursivní modelová struktura): Rekursivní modelová struktura RMS_{R_i} agenta R_i je definována jako dvojice

$$(P_{R_i}, RM_{R_i})$$

, kde P_{R_i} je matice zisků definovaná v 3.5 a RM_{R_i} je rekurzivní model 3.7, který je použit k modelování rozhodování ostatních agentů. ▶

Rekursivní model je definován v [4] takto:

Definice 3.7 (Rekursivní model): Rekursivní model MR_{R_i} agenta R_i je definován jako m-tice dvojic [1]

$$MR_{R_i} = ((p_1^{R_i}, M_{\{-R_i\}}^{(R_i,1)}), ..., (p_m^{R_i}, M_{\{-R_i\}}^{(R_i,m)}))$$

kde

$$M_{\{-R_i\}}^{(R_i,k)} = (M_{R_1}^{(R_i,k)}, ..., M_{R_{i-1}}^{(R_i,k)}, M_{R_{i+1}}^{(R_i,k)}, ..., M_{R_n}^{(R_i,k)})$$

představuje jednu z m $(n-1)$-tic rozhodovacích modelů ostatních agentů a p_i^k jeho subjektivní předpokládanou pravděpodobnost. ▶

$M_{R_j}^{(R_i,k)}$ je tedy jeden z možných modelů agenta R_j, který předpokládá agent R_i s pravděpodobností $p_k^{R_i}$. Platí samozřejmě podmínka $\sum_{k=1}^{m} p_k^{R_i} = 1$.
Model $M_{R_j}^{(R_i,k)}$ se dá podle [4] rozdělit do tří forem:

[1] $\{-R_i\}$ je zkrácený zápis množiny ostatních agentů $\{R_1, ..., R_n\}/\{R_i\}$

- $IM_{R_j}^{(R_i,k)}$ - Recionální model
- $NM_{R_j}^{(R_i,k)}$ - Neinformovaný model
- $SM_{R_j}^{(R_i,k)}$ - Neracionální model

Recionální model odpovídá tomu, že agent R_i předpokládá o agentovi R_j, že se chová racionálně. V [4] je definován jako

$$IM_{R_j}^{(R_i,k)} = RMS_{R_j}^{(R_i,k)}$$

,

s parametry $p_{a_l^j}^{(R_i,k)}$ a $P_{R_j}^{(R_i,k)}$. $P_{R_j}^{(R_i,k)}$ je matice zisků, kterou podle agenta R_i v modelu k agent R_j použije. což je Rekurzivní modelové struktura s k-tými daty, o kterých agent R_i předpokládá, že je agent R_j použije k rozhodování.

Neinformovaný model vychází z toho, že agent R_i nemá o agentovi R_j žádné informace, tudíž přiřadí každé akci $a_l^{R_j}$ pravděpodobnost $p_{a_l^j}^{(R_i,k)} = \frac{1}{|A_j|}$, kde $|A_j|$ je počet možných akcí agenta R_j, což odpovídá rovnoměrnému rozdělení. $p_{a_l^j}^{(R_i,k)}$ značí pravďepodobnost, že podle agenta R_i nastane v k-tém modelu agenta R_j akce a_l^j.

Neracionální model odpovídá tomu, že se agent A_j chová iracionálně. Chování se v tompto případě modeluje podle situace pokaždé jinak.

Systém se tedy rekurzivně rozvijí, dokud jsou dostupné informace. Pokud nejsou, rekurze skončí neinformovaným modelem s rovnoměrným rozdělením pravděpodobnosti všech akcí. Rekurze také může skončit iracionálním modelem, což ovšem není v našem případě příliš časté.

3.2.2 Rozhodovací algoritmus

Po zkonstruování rekurzivní modelové struktury se prochází tento systém od konce rekurze, kde jsou pravděpodobnosti známi, z důvodu ukončení rekurzivního rozvijení neinformovaným modelem. K výběru nejpravděpodobnější situace se používá tzv. užitečnosti. [4]

Definice 3.8 (Užitečnost): Užitečnost akce a_m^i podle agenta R_i je definována jako

$$u_{a_m^i}^{R_i} = \sum_{a_q^1 \in A_1} \dots \sum_{a_v^{i-1} \in A_{i-1}} \sum_{a_w^{i+1} \in A_{i+1}} \dots \sum_{a_x^n \in A_n} (p_{a_q^1}^{R_i} \dots p_{a_v^{i-1}}^{R_i} p_{a_w^{i-1}}^{R_i} \dots p_{a_x^n}^{R_i} u_{a_q^1 \dots a_m^i \dots a_x^n}^{R_i})$$

, kde

$$u_{a_q^1 \dots a_x^n}^{R_i}$$

je prvek matice zisků P_{R_i} a pravděpodobnost $p_{a_o^j}^{R_i}$ je definována jako

$$p_{a_k^j}^{R_i} = \sum_o p_o^{R_i} p_{a_k^j}^{(R_i,o)}$$

.

kde $p_o^{R_i}$ je pravděpodobnost modelu z definice 3.7 a $p_{a_k^j}^{(R_i,o)}$ značí pravďepodobnost, že podle agenta R_i nastane v o-tém modelu agenta R_j akce a_k^j. V případě, že je algoritmus v bodě racionálního modelu, určí se tato hodnota rekurzivně, pokud je model neinformovaný, je rovna $\frac{1}{|A_j|}$. $p_{a_k^j}^{R_i}$ je tedy součet pravděpodobností dané akce v modelu vyvážený pravděpodobností modelu a $u_{a_m^i}^{R_i}$ se definuje jako součet všech prvků matice zisků, kromě těch, které zahrnují jiné akce agenta R_i, než je a_k^j, vyvažený touto pravděpodobností.

3.3 Markovův rozhodvací proces

Markovův rozhodvací proces je alternativní metoda sloužící k volbě strategií odhadem zisků z nich plynoucích do budoucna. Modeluje prostředí jako množinu stavů a akcí, které je možné provést a mění stav prostředí s určitou pravděpodobností. Čas je zde vnímán v diskrétní formě kroků a prostředí jako částečně známé, částečně ovlivnitelné a do jisté míry měnící se náhodně.

Účelem tohoto procesu je najít funkci $\pi(s) : S \to A$ která každému stavu prostředí jednoznačně přiřadí akci maximalizující zisky do určitého bodu v budoucnosti, teoreticky do nekonečna. V [13] je Markovův rozhodovací proces definován takto:

Definice 3.9 (Markovův rozhodovací proces): Markovův rozhodovací proces je definován jako uspořádaná čtveřice

$$(S, A, P, R)$$

kde jsou

- S - množina stavů prostředí
- A - množina možných akcí
- $P(i, a, j) := p(s_{t+1} = j | s_t = i \wedge a = a_t)$ - pravděpodobnost, že při aplikování akce a za stavu i přejde systém do stavu j
- $R(i, a, j)$ - okamžitý užitek přechodu systému ze stavu i do stavu j při akci a ▶

3.3.1 Dynamické programování

Dynamické programování je proces sloužící k nalezení optimální strategie dané $\pi(s)$. K tomu slouží funkce očekávaných kumulativních diskontních zisků $V^\pi(i) :\rightarrow \mathbb{R}$ reprezentující předpokládané dlouhodobé zisky při určité $\pi(s)$. Agent se tedy snaží zvolit strategii tak, aby funkci V^π při daném počátečním stavu $s_0 = i$ maximalizoval. V [13] je tato funkce definována následovně:

Definice 3.10 (V-funkce):

$$V^\pi(i) = \sum_{k=0}^{\infty} E(\gamma^k R(s_k, \pi(s_k), s_{k+1}))$$

při $s_0 = i$ kde jsou

- $\gamma \in < 0, 1 >$ diskontní faktor určující snižování významnosti odhadu s narůstajícím časem
- E operator odhadu ▶

Tento zápis je spíše formální, proto se tato funkce v [13] definuje pomocí pomocí Q-funkce $Q^\pi(i, a)$, funkce odhadující zisk při aplikování akce a při stavu a.

Definice 3.11 (Q-function):

$$Q(i, a)^\pi = \sum_{j \in S} P(i, a, j)(R(i, a, j) + \gamma V^\pi(j))$$

kde jsou

$$V^\pi(i) = \max_a Q^\pi(i, a)$$

,

$$\pi(i) = \arg\max_a Q^\pi(i, a)$$

Postupným iterativním přepočítáváním V, Q a π dostaneme Bellmanovu rovnici optimality definovanou v [2] jako:

Definice 3.12 (Bellmanova rovnice optimality):

$$Q(i,a)^* = \sum_{j \in S} P(i,a,j)(R(i,a,j) + \gamma V^*(j))$$

kde funkce Q a V nezávisí na strategii π. Bellmanova rovnice optimality říká, že pokud jsou Q a V optimální, nemění se s dalšími kroky iterace. V praxi se stanoví nějaká dostatečně malá konstanta ϵ, a pokud se v dalším kroku Q a V nezmění o vícejak ϵ, považujeme je za optimální. Navíc bylo dokázáno, že vždy existuje právě jedna dvojice dvojice těchto funkcí, která je optimální a vždy ji lze tedy iterací určit.

Algoritmus pro určení funkce π tedy iterativně přes všechny stavy, akce a časové kroky upravuje funkce V, Q a π podle rovnic 3.11, a v [13] je v krocích popsán takto:

1. Inicializace počátečních hodnot funkcí V a Q
2. Opakovat kroky 3-5 do dosažení podmínek optimality
3. Upravit hodnotu Q-funkce podle rovnice
$$Q(i,a) = \sum_{j \in S} P(i,a,j)(R(i,a,j) + \gamma V(j))$$
4. dopočítat hodnotu funkce V jako
$$V(i) = \max_a Q(i,a)$$
5. nastavit hodnotu funkce pi na akci maximálního zisku jako
$$\pi(i) = \arg\max_a Q(i,a)$$

3.4 Zpětnovazebné učení (Reinforcement learning)

V praxi jsou z počátku hodnoty funkcí P a R popsané v 3.9 neznámé a je potřeba je určit pozorováním. Prakticky se tedy agent musí naučit jak prostředí reaguje na provedení každé akce v určitém stavu. Podle [13] se tedy na začátku procesu nastaví všechny hodnoty na nulu a po každé provedené akci se pozoruje jak prostředí reaguje, tedy hodnota funkce $P(i,a,j)$, a jaký okamžitý zisk z toho plyne, tedy $R(i,a,j)$. K určení těchto hodnot se používají různé algoritmy.

3.4.1 Q-učení (Q-learning)

Q-učení je bezmodelová metoda popsaná v [12] a [8]. K určení požadovaných hodnot, jak název napovídá, používá úpravu funkce Q definované v 3.11 v každém časovém kroce t podle rovnice:

$$Q_t(i,a) = \begin{cases} (1-\alpha_t)Q_{t-1}(i,a) + \alpha_t(r_t + \gamma V_{t-1}(i)) & : i = i_t \wedge a = a_t \\ Q_{t-1}(i,a) & : jinak \end{cases}$$

kde $\alpha_t \in <0,1>$ je učící faktor v časovém kroce t a r_t je okamžitý zisk akce a. Funkce P a R popisující prostředí se zde tedy nevyskytují a Q-funkce se zde upravuje přímo z naměřených hodnot za použití snižujícího se učícího koeficientu α.

3.4.2 Učení na základě modelu (Model-based learning)

V této metodě, popsané v [13], se modeluje prostředí funkcemi $P(i,a,j)$ a $R(i,a,j)$, které jsou definované v 3.9. Ty jsou z počátku neznámé a jejich hodnoty se určují metodou maximální věrohodnosti podle četnosti naměřených údajů. Agent popsaný v [13] používal následující veličiny:

- $C_i(a)$ - počet, kolikrát agent zvolil akci a ve stavu i
- $C_{i,j}(a)$ - počet, kolikrát prostředí přešlo ze stavu i do stavu j při aplikace akce a
- $R_{i,j}(a)$ - součet okamžitých zisků při použití akce a ve stavu i a následném přechodu do stavu j

Model maximální věrohodnosti (MLM) je v [13] definován jako:

Definice 3.13 (MLM):

$$\hat{P}(i,a,j) = \frac{C_{i,j}(a)}{C_i(a)}$$

$$\hat{R}(i,a,j) = \frac{R_{i,j}(a)}{C_{i,j}(a)}$$

V každém časovém kroce je po aplikaci akce a model přepočítán. Poté je znovu použito dynamické programování popsané v sekci 3.3.1.

3.5 Bayesovské učení

V této kapitole je naznačena metoda bayesovského učení , což je bodový odhadu parametru rozdělení náhodné veličiny založený na Bayesovuě větě. Tato metoda byla použita v článku [4] ke zlepšení odhadu pravděpodobnosti uskutečnění se modelového chování ostatních agentů. Jeho výhodou je využití experimentální i expertní znalosti. V praxi se této metody hojně využívá tam, kde jsou data získávána postupně a zpočátku je nutné parametr odhadnout na základě zkušenosti a nadále ho podle výsledků měření zpřesňovat.

Definice 3.14 (Podmíněná pravděpodobnost): Nechť A a B jsou náhodné jevy. Podmíněná pravděpodobnost jevu A na jevu B, tedy pravděpodobnost že nastane jev A pokud nastane jev B, je definována jako

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

Věta 3.1 (Bayesova věta): *Nechť A a B jsou náhodné jevy a platí $P(B) > 0$. Potom*

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

V souladu s [5] zavedeme následující značení:

- $y(t) = [y_1, ..., y_T]$ - vetor $T \in \mathbb{N}$ naměřených hodnot, kde $y_t \in \{0, 1\}$ a $y_t = 1$ značí, že výsledek experimentu v čase $t \in \{1, ..., T\}$ dopadlu v souladu s naší hypotézou
- Θ - parametr rozdělení
- $\hat{\Theta}_T$ - odhad parametru při T naměřených hodnotách
- $f(y_t|\Theta)$ - hustota pravděpodobnosti náhodné veličiny v čase t při parametru Θ

3.5.1 Věrohodnostní funkce

K odhadu parametru se v této metodě využívá věrohodnostní funkce, která vyjadřuje pravděpodobnost, že při daném parametru se bude vektor naměřených výsledků bude rovnat teoretické hodnotě. V [5] se tato funkce definuje jako: [2]

[2]Tato definice je poněkud zjednodušena a předpokládá, že výsledek t-tého pokusu nezávisí na předchozích pokusech, ani na počátečních podmínkách před t-tým pokusem.

Definice 3.15 (Věrohodnostní funkce):

$$f(y(T)|\Theta) = \prod_{t=1}^{T} f(y_t|\Theta)$$

Při použití těchtu veličin nabývá Bayesova věta podle [5] tvaru

$$f(\Theta|y(T)) = \frac{f(y(T)|\Theta)f(\Theta)}{f(y(T))}$$

kde jsou

- $f(\Theta|y(T))$ aposteriorní hustota pravděpodobnosti
- $f(y(T)|\Theta)$ Věrohodnostní funkce
- $f(\Theta) = f(\Theta|y(0))$ je tzv. apriorní hustota pravděpodobnosti
- $f(y(T))$ normalizační konstanta ($f(\Theta|y(T))$ se bere jako funkce Θ a parametrů $y(t)$)

Apriorní hustota pravděpodobnosti zde vyjadřuje odhad parametru při nenaměřených datech. Jedná se tedy o expertní odhad, předpokládané chování náhodné veličiny založené na známých faktech a zkušenosti. Aposteriorní hustota pravděpodobnosti představuje pravděpodobnost parametru Θ závislou na naměřených hodnotách náhodné veličiny a její střední hodnota je odhadem tohoto parametru. Tedy platí:

$$\hat{\Theta}_T = \int_{\mathbb{R}} f(\Theta|y(T))d\Theta$$

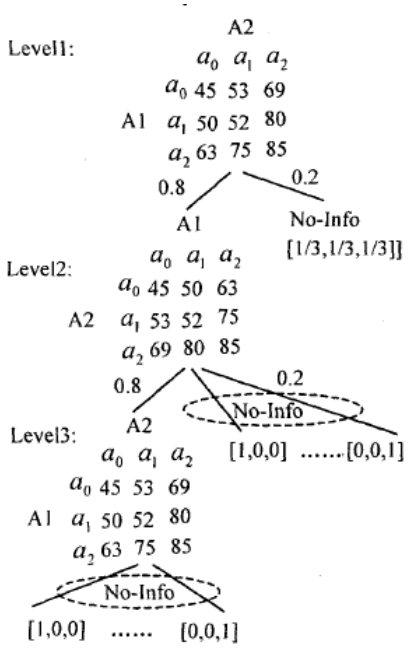
3.6 Použití RMM a Bayesova učení v decentralizovaném řízení dopravy

V následujícím textu se budeme zabývat konkrétním využitím rekursivních modelovacích metod v kombinaci s bayesovským učením v decentralizovaném řízení dopravy pomocí multiagentních systémů. Tento koncept byl testován a výsledky zveřejněny v článku [6].

Testy byly prováděny na modelu dvou křižovatek klasického typu, kde každý agent R_1, R_2 ovládal jednu z nich. K modelování chování druhého agenta bylo použito třísyupňové RMM zakončené neinformovaným modelem. Obrázek 3.1 z článku [6] znázorňuje RMM

agenta R_1, kde se na každé úrovni agent rozhoduje mezi dvěma modely, informovaným a neinformovaným. Z počátku simulace jsou těmto modelům přiřazeny pravděpodobnosti po řadě 0,8 a 0,2. Tyto pravděpodobnosti jsou upravovány metodou bayesovskeho učení podle akcí provedených druhým agentem. Pro řešení tohoto modelu je použit algoritmus z [4] popsaný v předchozí kapitole.

Určujícím faktorem pro sestavení matice zisků je délka fronty čekajících aut. Tento údaj byl také použit jako klíčový pro srovnání s ostatními metodami při testování. Metoda byla testována za použití simulace psané v jazyce VC++ a porovnávána s dvěma dalšími, a to pevně dané sekvence stanovené dopravními experty a stochastická metoda "Hill-climbing process". Z tohoto srovnání vychází metoda využívající RMM a bayesovského učenís podle výsledků prezentovaných v [6] nejlépe, jak můžeme vidět na obrázku 3.2.



Obrázek 3.1: Tříúrovňová rekurzivní modelová struktura agenta R_1

Traffic flow entered Our Experimental Network (vehicle/hour)	Average waiting car's queue during each cycle		
	M1	M2	M3
<1008	<0.05	<0.05	<0.05
1008	0.1	0.12	0.19
1152	0.15	0.17	0.24
1296	0.26	0.29	0.37
1368	0.34	0.48	0.75
1512	0.47	0.80	1.12
1656	0.77	1.08	1.95
1728	0.91	1.67	2.54
1944	1.23	2.10	3.38
2088	1.94	2.45	5.80
2186	2.58	3.87	6.78
2320	3.56	4.72	8.92
2548	4.03	6.43	11.2
2670	4.85	8.77	16.56
2858	5.60	10.27	*
3016	6.48	13.89	*
3120	7.86	15.73	*
3368	9.22	17.23	*
3420	10.26	*	*
3608	11.22	*	*

Obrázek 3.2: Výsledky měření. M1 - RMM, M2 - pevné cykly, M3 - "Hill-climbing"

3.7 Použití zpětnovazebného učení

3.7.1 Zpětnovazebné učení na základě modelu

V [13] je popsána simulace používající zpětnovazebné učení na základě modelu popsané v 3.4.2. Na rozdíl od ezperimentu popsaného v předchozí kapitole zde autoři zvolili variantu, kde existují 2 druhy agentů: agent-vozidlo a agent-křižovatka. Agent-vozidlo má v každý časový krok následující parametry definující jeho stav, které se ukládají v průběhu celé simulace:

- *node* - dopravní uzel, kde se právě nachází
- *dir* - směr vůči tomuto uzlu
- *place* - pozice ve frontě vozidel
- *des* - místo, kam chce v systému dojet

Algoritmus se snaží minimalizovat celkový čas od vyjetí až po dosažení cíle. Každý agent má ve vyjednávání právo hlasovat o nastavení dopravního uzlu, tedy jestli bude v průjezdném nebo uzavřeném stavu. Pro každý uzel *node* a směr *dir* je to tedy akce g - na uzlu *node*, kde se vozidlo nachází ve směru *dir*, svítí červená - uzel je v tomto směru neprůjezdný, a akce g - uzel je v tomto směru průjezdný. Funkce definované v 3.3.1 se tedy zapisují ve tvarech

$$Q([node, dir, place, des], a)$$

$$V([node, dir, place, des])$$

kde $a \in \{r, g\}$. Optimální akce se v [13] volí podle maxima sumy z

$$Q([node, dir, place, des], r) - Q([node, dir, place, des], g)$$

přes všechny vozidla ovlivněná frontou u daného uzlu. Komunikace je zde tedy omezena pouze na výměnu hodnot Q.

Metode popsaná v článku [13] používá ohodnocovací funkci založenou na parametrech jednotlivých vozidel. Výhodou oproti pojetí, kdy agent představuje pouze signální skupinu, jsou například v tom, že není potřeba odhadovat délku fronty a úloha se celá zjednoduší. Například v publikaci [9] se musí používat k odhadu funkcí V a Q neuronová síť. Navíc tento systém umožňuje i výběr optimální cesty vozidla pro průjezd dopravní sítí. Nevýhodou tohoto pojetí je ovšem značná neuniverzálnost. Už pro počítačové testování tato metode znesnadnuje či úplně znemožňuje použít celou řadu dopravních simulátorů, které jsou pro simulaci po dlouhou dobu optimalizovány a jejichž nasazení značně zjednodušuje práci a urychluje vývoj. Navíc pokud je použit řadič, který obstarává logiku přepínání průjezdnosti a lze nastavovat pouze vnější parametry jako jsou délka cyklu a offset, je metoda, která potřebuje okamžitou změnu signalizace naprosto nevhodná, proto je toto řešení pro reálné nasazení v dnešní době nepoužitelné.

3.8 LQ řízení

LQ je zkratka slovního spojení linaer-quadratic, tedy lineárně kvadratický. Jedná se obecně o metodu, kdy je systém v diskrétním časovém kroku t popsán vektorem proměnných $x(t) = (x_1(t), ..., x_n(t))$ a my můžeme nastavovat vektor parametrů $y(t) = (y_1(t), ..., y_n(t))$.

Metoda je popsána a použita v článku [10] pro nastavování časů zelených na křižovatce. Princip přechodu z času t do $t+1$ je popsán lineárním vztahem

$$x(t+1) = Ax(t) + By(t) \tag{3.1}$$

. kde matice A a B jsou matice stavů a vstupů vyjadřující odezvu systému vzhledem k $x(t)$ a $y(t)$. Účelem LQ řízení je najít optimální hodnoty $y(t)$ v závislosti na $x(t)$ dané zpětnovazebnou maticí L vztahem

$$y(t) = -Lx(t) \tag{3.2}$$

. Optimalita je definována pomocí kvadratického kritéria

$$J = \sum_{t=1}^{\infty} x(t)^T Q x(t) + y(t)^T R y(t) \tag{3.3}$$

, kde Q a R jsou diagonální pozitivně semidefinitní matice vah určující významnost jednotlivých členů kritéria. Zpětnovazebná matice L se podle [10] dostane minimalizací kritéria J jako

$$L = (R + B^T P B)^{-1} B^T P A \tag{3.4}$$

, kde P je jednoznačné řešení Riccatiho rovnice pro diskrétní časový krok

$$P = Q + A^T (P - PB(R + B^T P B) B^T P) A \tag{3.5}$$

. Rovnici a její řešení popisuje například [1].

3.8.1 Použití LQ řízení ve strategii TUC

LQ řízení bylo použito v [10] k nalezení optimální délky zelených v systému 13-ti signálních skupin. Proměnné $x_i(t)$ zde představují obsazenost ramene i spojující křižovatky M a N. Účelem strategie je nalezení optimální délky zelených g, $g_{N,i}$ značí délku zelené na signální skupiny křižovatky N zprůjezdňující směr do ramene i. Předpokládaný vztah pro přechod systému z času t do času $t+1$ je

$$x_i(t+1) = x_i(t) + T[q_i(t) + s_i(t) + d_i(t) + u_i(t)] \tag{3.6}$$

, kde proměnné značí:

- T - časový krok

- $q_i(t)$ - přírůstek vozidel z křižovatek
- $u_i(t)$ - úbytek vozidel do ostatních křižovatek
- $s_i(t)$ - přírůstek vozidel z okolí sítě
- $d_i(t)$ - úbytek vozidel mimo síť

Přírůstek vozidel z křižovatek je dán vztahem

$$q_i(t) = \sum_{k \in I_m} t_{k,i} u_k(t) \tag{3.7}$$

, je to tedy součet úbytků vozidel z ramen ústících do křižovatky N vynásobených koeficinety $t_{k,i}$, což jsou odbočovací poměry z ramene k do ramene i. V podovném tvaru se předpokládá $s_i(t)$

$$s_i(t) = t_{i,0} q_i(t) \tag{3.8}$$

, kde $t_{i,0}$ je odbočovací koeficient ramene i mimo sledovanou síť. Při délce cyklu C, saturovaném toku S_i a délce zelených $g_{N,i}(t)$ ramene i platí

$$u_i(t) = \frac{S_i \sum g_{N,i}(t)}{C} \tag{3.9}$$

. Rovnice 3.6 tedy přechází do tvaru

$$x_i(t+1) = x_i(t) + T \left[(1 - t_{i,0}) \sum_{k \in I_m} t_{k,i} \frac{S_k \sum g_{M,k}(t)}{C} - \frac{S_i \sum g_{N,i}(t)}{C} + d_i(t) \right] \tag{3.10}$$

. Uvažujeme-li nominální hodnoty d^n a g^n vedoucí vždy na stav x^n, platí podle rovnice 3.10

$$0 = T \left[(1 - t_{i,0}) \sum_{k \in I_m} t_{k,i} \frac{S_k \sum g^n_{M,k}}{C} - \frac{S_i \sum g^n_{N,i}}{C} + d^n_i \right] \tag{3.11}$$

. Označíme-li

$$\Delta g(t) = g(t) - g^n \tag{3.12}$$

, můžeme psát rovnici 3.10 jako

$$x_i(t+1) = x_i(t) + T \left[(1 - t_{i,0}) \sum_{k \in I_m} t_{k,i} \frac{S_k \sum \Delta g_{M,k}(t)}{C} - \frac{S_i \sum \Delta g_{N,i}(t)}{C} \right] \tag{3.13}$$

, což dovoluje tuto rovnici zapsat pomocí matic v požadovaném tvaru

$$x(t+1) = Ax(t) + B\Delta g(t) \tag{3.14}$$

, kde A je jednotková matice.

3.8.1.1 Kvadratické kritérium

Účelem lagoritmu je minimalizovat obsazenost ramen, tedy vektor $x(t)$ a penalizovat změnu délky trvání zelené oproti nominálním hodnotám. Kvadratické kritérium optimálního řízení 3.3 jetedy v [10] definováno vztahem

$$J = \sum_{t=0}^{\infty} x(t)^T Q x(t) + \Delta g(t)^T R \Delta g(t) \tag{3.15}$$

. Diagonální matice Q je zde zodpovědná za vyvažování počtu vozidel jednotlivých úseků. V [10] je každý diagonální prvek $Q_{i,i}$ matice Q položen převrácené hodnotě maximálního povoleného počtu vozidel daného úseku i. $R = rI$ penalizuje změnu časů zelených. Parametr r ovlivňuje míru reakce systému a ja volen metodu pokus-oprava. Minimalizací tohoto kritéria pomcí 3.4 získáme zpětnovazebnou matici L, která určuje $g(t)$. Z rovnic 3.2 a 3.12 dostaneme výsledný vztah

$$g(t) = g^n - Lx(t) \tag{3.16}$$

. Toto řešení předpokládá, že známe hodnotu g^n, při které systém zůstává ve stavu x^n. Tak tomu ale většinou není. Při absenci znalosti g^n podle [10] odečteme $g(t) - g(t-1)$ a rovnice 3.16 nabývá tvaru

$$g(t) = g(t-1) - L(x(t) - x(t-1)) \tag{3.17}$$

.

Kapitola 4

Implementace

4.1 Seznam proměnných

Všechny proměnné jsou zavedeny pro každého agenta reprezentujícího jednu křižovatku zvlášť a jsou označeny indexem i signální skupiny. Tavždy ovládá jeden jízdní pruh. Pokud jsou bez indexu, jedná se o proměnnou agenta, případně oblasti simulace.

V textu	V programu	Název	Popis
$T \in \mathbb{N}$	`T`	Vzorkovací perioda	...
$C \in \mathbb{N}$	`Tc`	Délka cyklu	...
$S = 0,5$	`Tc`	Saturovaný tok	Maximální počet vozidel, která projedou křižovatkou za sekundu
$L \in \mathbb{R}^+$	`L`	Ztrátový čas	Čas potřebný k vyklizení křižovatky mezi dvěma fázemi
$J = \{j_1, ..., j_n\}$		Množina signálních skupin	
$J_A \subset J$		Množina signálních skupin agenta A	Skupiny náležící jednomu agentovi
$I_j \subset J$		Množina vstupů signální skupiny	Signální skupiny ostatních agentů ústících do příslušné skupiny g
$g_j^N \in < 0, 1 >$		Nominální poměr zelené	Defi novaný poměr fází
$g_j(t) \in < 0, 1 >$		Efektivní poměr zelené	Část z délky cyklu, kdy je signální skupina průjezdná
$i_j(t) \in \mathbb{R}^+$	`i_g`	Hustota vstupů	Počet přijíždějících vozidel do pruhu signální skupiny za jednotku času
$o_j(t) \in \mathbb{R}^+$		Hustota výstupů	Počet vyjíždějících vozidel z pruhu signální skupiny za jednotku času
$q_j(t) \in \mathbb{N}_0^+$		Fronta	Počet vozidel jedoucí menší rychlostí než $3,6km/h$
$\alpha_{j,k} \in < 0, 1 >$		Odbočovací poměr z j do k	

4.2 Použitá metoda

Účelem této práce je řídit provoz pomocí nastavení optimální délky cyklu za použití multiagentního systému, kde každý agent ovládá signální skupiny jedné křižovatky a jsou mu dostupné příslušné údaje. Jako vhodná metoda bylo zvoleno LQ řízení, které se snaží minimalizovat fronty a penalizovat délku cyklu.

4.2.1 Přechodové vztahy

Jako údaj popisující stav systému byla zvolena délka fronty $q_j(t)$, což je počet aut v jízdním pruhu jedoucích méně než 3,6 km/h. Pro danou frontu v čase $t+1$ platí zřejmě vztah

$$q_j(t+1) = q_j(t) + T(i_j(t) - o_j(t)) \tag{4.1}$$

, kde hustota vstupu $i_j(t)$ je součtem výstupů sousedů pronásobený odbočovacími poměry, případně vstupu do systému $i_{j,0}(t)$, pokud se jedná o koncové rameno řízené oblasti, tedy

$$i_j(t) = i_{j,0}(t) + \sum_{k \in I_j} \alpha_{k,j} o_k(t) \tag{4.2}$$

. Křižovtkou z daného jízdního pruhu v průjezdné fázi projede S vozidel za sekundu, kde délka průjezdné fáze závisí na nastaveném poměru a délce cyklu. Před přechodem do nové fáze signálího plánu je potřeba vklidit křižovatku, což vede ke ztrátovému času L, který se projeví vždy jednou za cyklus. Efektivní délka zelené je tedy

$$g_j(t) = (C(t) - L)g_j^N \tag{4.3}$$

a pro hustotu výstupu platí

$$o_j(t) = \frac{S_j g_j(t)}{C(t)} = S_j g_j^N \left(1 - LC(t)^{-1}\right) \tag{4.4}$$

. Rovnice 4.1 tedy přechází do tvaru

$$q_j(t+1) = q_j(t) + T\left(i_{j,0}(t) + \sum_{k \in I_j} \alpha_{k,j} S_k g_k^N \left(1 - LC(t)^{-1}\right) - S_j g_j^N \left(1 - LC(t)^{-1}\right)\right) \tag{4.5}$$

. Pokud budeme podobně jako v [10] předpokládat existenci nominální délky cyklu C^N, při níž se nemění délka fronty, platíc(

$$0 = T\left(i_{j,0}(t) + \sum_{k \in I_j} \alpha_{k,j} S_k g_k^N \left(1 - LC^{N-1}\right) - S_j g_j^N \left(1 - LC^{N-1}\right)\right) \tag{4.6}$$

. Odečtením rovnice 4.6 od 4.5 dostaneme

$$q_j(t+1) = q_j(t) + T\left(-\sum_{k \in I_j} \alpha_{k,j} S_k g_k^N + S_j g_j^N\right) L\Delta C^{-1}(t) \tag{4.7}$$

, kde $\Delta C^{-1}(t) = C(t)^{-1} - C^{N-1}$. Pro ramena $j \in J_A$ náležící agentovi A se tedy rovnice pro fronty dají zapsat maticově ve tvaru

$$\vec{q}(t+1) = A\vec{q}(t) + B\Delta C^{-1}(t) \tag{4.8}$$

Kostantni saturovany tok neni optimalni, pouzije aprox podle Pecherkove (viz bak.), prujezd odpovida bud sat. toku, pokud neodjede cela fronta, nebo je primo umerny fronte (+ prijezdum). prolozi se funkci a aprox se primkou podle hodnoty fronty v case t. Napocte se rizen pro horizont a podle vyvoje fronty se prepocte misto pro prolozeni pri

4.2.1.1 Model toku

V článku [10] se úbytek vozidel modeluje podle rovnice 3.9, kde je uvažován tok vozidel křižovatkou za jednotku času jako konstanta S, což je saturovaný tok. Tento vztah platí pouze když je příslušný pruh naplněn vozidly do té míry, že křižovatkou projede maximální možný počet vozidel. To však neplatí, pokud není pruh dostatečně vytížen, například pokud se zmenší délka fronty v nějaký časový okamžik na nulu. Pokud je vytížení menší je tok lineárně závislý na součtu fronty $q(t)/T$ a počtu vstupujících vozidel $i(t)$. Pro teoretickou hodnotu toku S tedy dostáváme vztah

$$S_{teor}(q(t)+i(t)) = \begin{cases} \frac{q(t)}{T} + i(t) & \frac{q(t)}{T} + i(t) <= S_{max} \\ S_{max} & \frac{q(t)}{T} + i(t) > S_{max} \end{cases} \tag{4.9}$$

. tento vztah vyjadřuje to, že pokud je fronta plus přírůstrek vozidel ve vzorkovací periodě $(q(t), i(t)T)$ menší než maximální počet vozidel, který je křižovatka za periodu T schopna propustit (TS_{max}), projedou všechna vozidla. Abychom se vyhnuli podmínce podle S_{max}, aproximuje se tento vztah funkcí

$$S_{exp}(q(t), i(t)) = S_{max}\left(1 - e^{-\frac{1}{S_{max}}\left(\frac{q(t)}{T} + i(t)\right)}\right) \tag{4.10}$$

, která splňuje podmínku

$$\frac{dS_{exp}}{d(q/T+i)}(0,0) = 1 \tag{4.11}$$

, tedy při malé frontě a hustotě provozu je přírůstek toku roven $q/T + i$, a podmínku

$$\lim_{q/T+i\to\infty} S_{exp} = S_{max} \tag{4.12}$$

, tedy při velké frontě a hustotě provozu se tok blíží konstantě S_{max}. Pro malý časový interval můžeme tuto funkci dále zjednodušit na lineární vztah

$$S(q(t)) = \sigma(q(t)/T + i_0) \tag{4.13}$$

, kde je $i_0 = i(0)$ a

$$\sigma = \frac{dS_{exp}}{d(q/T+i)}(q(0), i(0)) \tag{4.14}$$

vlozit graf s ruznymi S

4.2.1.2 Přechodové vztahy s proměnným tokem

Pro vyjádření přechodových vzthů použijeme lineární odel toku a rovnice 4.5 přejde na tvar

$$q_j(t+1) = q_j(t) + T\left(i_{j,0}(t) + \sum_{k \in I_j} \alpha_{k,j} S_k(q_k(t)) g_k^N \left(1 - LC(t)^{-1}\right) - S_j(q_j(t)) g_j^N \left(1 - LC(t)^{-1}\right) \right) \tag{4.15}$$

, a po dosazení 4.13 dostaneme

$$q_j(t+1) = q_j(t) + T \begin{pmatrix} i_{j,0}(t) + \\ + \sum_{k \in I_j} \alpha_{k,j} \delta_k (q_k(t) T^{-1} + i_{0,k}) g_k^N \left(1 - LC(t)^{-1}\right) - \\ -\delta_j (q_j(t) T^{-1} + i_{0,j}) g_j^N \left(1 - LC(t)^{-1}\right) \end{pmatrix} \tag{4.16}$$

$$q_j(t+1) = q_j(t) + T \begin{pmatrix} i_{j,0}(t) + \\ + \sum_{k \in I_j} \alpha_{k,j} \delta_k g_k^N T^{-1} q_k(t) - \delta_j g_j^N T^{-1} q_j(t) + \\ + \sum_{k \in I_j} \alpha_{k,j} \delta_k i_{0,k} g_k^N - \delta_j i_{0,j} g_j^N - \\ - \sum_{k \in I_j} \alpha_{k,j} \delta_k g_k^N L T^{-1} q_k(t) C(t)^{-1} + \delta_j g_j^N L T^{-1} q_j(t) C(t)^{-1} - \\ - \sum_{k \in I_j} \alpha_{k,j} \delta_k i_{0,k} g_k^N L C(t)^{-1} + \delta_j i_{0,j} g_j^N L C(t)^{-1} \end{pmatrix} \tag{4.17}$$

. Pokud budeme podobně jako v rovnici 4.6 uvažovat nominální C^N, pro které se $q(t+1)$ rovná $q(t)$, dostaneme dosazením $\Delta C^{-1}(t) = C(t)^{-1} - C^{N-1}$ do 4.17 rovnici

$$\begin{aligned} q_j(t+1) \quad = q_j(t) + \quad & L\left(\delta_j g_j^N q_j(t) - \sum_{k \in I_j} \alpha_{k,j} \delta_k g_k^N q_k(t)\right) C(t)^{-1} + \\ & + LT\left(\delta_j i_{0,j} g_j^N - \sum_{k \in I_j} \alpha_{k,j} \delta_k i_{0,k} g_k^N\right) C(t)^{-1} \end{aligned} \tag{4.18}$$

4.2.2 Minimalizační kritérium

Podobně jako v předchozí kapitole, minimalizovat kvaratické kritérium J. Minimalizaci budeme provádět v proměnných $\Delta C(t)$ pro časový horizont h, tedy pro vektory $\Delta C(t_0), \Delta C(t_0 + 1), ..., \Delta C(t_0 + h)$. Pro zpřehlednění zápisu položíme bez újmy na obecnosti $t_0 = 0$. Kvadratické kritérium ve tvaru

$$J = \sum_{t=0}^{h} q(t)^T Q q(t) + \Delta C(t)^T R \Delta C(t) \tag{4.19}$$

, kde Q a R jsou diagonální pozitivně definitní matice vah, rozepíšeme pomocí vztahu

$$q(t+1) = Aq(t) + Bq(t) \tag{4.20}$$

a budeme minimalizovat pro jednotlivá t podle $\Delta C(t)$. $\Delta C(h)$ se v kritériu vyskytuje pouze jako člen

$$\Delta C(h)^T R \Delta C(h) \tag{4.21}$$

a tedy musí být $\Delta C(h)0$. $\Delta C(h-1)$ se vyskytuje ve členech

$$\Delta C(h-1)^T R \Delta C(h-1) + (q(h-1)^T A^T + \Delta C(h-1)^T B^T) Q (Aq(h-1) + B\Delta C(h-1)) \tag{4.22}$$

, což lze pomocí složených vektorů a matic přepsat do tvaru

$$(\Delta C(h-1)^T, q(h-1)^T) \begin{pmatrix} B^T\sqrt{Q} & \sqrt{R} \\ A^T\sqrt{Q} & 0 \end{pmatrix} \begin{pmatrix} \sqrt{Q}B & \sqrt{Q}A \\ \sqrt{R} & 0 \end{pmatrix} \begin{pmatrix} \Delta C(h-1) \\ q(h-1) \end{pmatrix} \tag{4.23}$$

, kde $\sqrt{Q}$ je matice, pro níž platí $\sqrt{Q}\sqrt{Q} = Q$. Složenou matici

$$\begin{pmatrix} B^T\sqrt{Q} & \sqrt{R} \\ A^T\sqrt{Q} & 0 \end{pmatrix} \begin{pmatrix} \sqrt{Q}B & \sqrt{Q}A \\ \sqrt{R} & 0 \end{pmatrix} = \begin{pmatrix} B^TQB + R & B^TQA \\ A^TQB & A^TQA \end{pmatrix} \tag{4.24}$$

můžeme pomocí Choleského dekompozici na součin dolní a horní trojúhelníkové matice. Člen 4.23 poté přejde na tvar

$$(\Delta C(h-1)^T, q(h-1)^T) \begin{pmatrix} L_q^T & 0 \\ L^T & L_C^T \end{pmatrix} \begin{pmatrix} L_C & L \\ 0 & L_q \end{pmatrix} \begin{pmatrix} \Delta C(h-1) \\ q(h-1) \end{pmatrix} \tag{4.25}$$

, který můžeme dále upravit na

$$(L_C\Delta C(h-1) + L_q q(h-1))^T (L_C\Delta C(h-1) + L_q q(h-1)) + q(h-1)^T L_q^T L_q q(h-1) \tag{4.26}$$

. Pokud tento člen chceme minimalizovat v proměnné $q(h-1)$, musí nutně platit

$$\Delta C(h-1) = -L_C^{-1} L q(h-1) \tag{4.27}$$

. Zbývající nenulová část $q(h-1)^T L_q^T L_q q(h-1)$ lze přepsat pomocí rovnice 4.20 do tvaru

$$q(h-1)^T L_q^T L_q q(h-1) = (Aq(h-2) + B\Delta C(h-2))^T L_q^T L_q (Aq(h-2) + B\Delta C(h-2)) \tag{4.28}$$

, kde se vyskytuje $\Delta C(t)$ pouze pro $t = h-2$. Minimalizace přes $\Delta C(h-2)$ bude probíhat analogicky, pouze se matice Q nahradí maticí

$$\widetilde{Q} = Q + L_q^T L_q \tag{4.29}$$

4.3 Simulace

4.3.1 Simulátor AIMSUN

4.3.2 Oblast simulace

Kapitola 5

Výsledky

Literatura

[1] Anderson, M. J., B.D.O.: Optimal Control - Linear Quadratic Methods. *Prentice Hall, Englewood Cliffs, NJ*, 1990.

[2] Bellman, R.: Dynamic programming. *Princeton University Press*, 1957.

[3] Ferreira, E.; Subrahmanian, E.; Manstetten, D.: Intelligent agents in decentralized traffic control. *Intelligent Transportation Systems*, 2001.

[4] Gmytrasiewicz, P. J.; Durfee, E. H.: A rigorous, operational formalization of recursive modeling. *First International Conference on Multiagent Systems*, 1995.

[5] Nagy, I.; Nedoma, P.; Ettler, P.; aj.: O bayesovském učení. *Automa*, 2002.

[6] Ou, H.; Zhang, W.; Xu, X.: Urban traffic multi-agent system based on RMM and Bayesian learning. *American Control Conference*, 2002.

[7] P., P.; J., D.; Flídr: Modelling and Simultaneous Estimation of State and Parameters of Traffic System. *Robotics, Automation and Control*, 2008.

[8] Sutton, R. S.: Learning to predict by the methods of temporal didffrences. *Machine Learning*, 1988.

[9] Thorpe, T.: Vehicle traffic light controlusing sarsa. *Master's thesis, Department of Computer Science, Colorado State University*, 1997.

[10] Vaya Dinopoulou, M. P., Christina Diakaki: Applications of the urban traffic control strategy TUC. *European Journal of Operational Research*, 2005.

[11] Watkins, C. J. C. H.: Leraning from Delayed Rewards. *PhD thesis, King's College, Cambridge, England*, 1989.

[12] Watkins, C. J. C. H.; Dayan, P.: Q-leraning. *Machine Learning*, 1992.

[13] Wiering, M.; Van Veenen, J.; Vreeken, J.; aj.: Intelligent traffic light control. *European Research Consortium for Informatics and Mathematics*, 2003.

[14] Wooldridge, M.: *Multi Agent Systems*. MIT Press, Březen 2005.

Příloha A

Příloha 1