INŽENÝRSKÁ MECHANIKA 2005
NÁRODNÍ KONFERENCE
s mezinárodní účastí
Svratka, Česká republika, 9. - 12. května 2005

MOBILE ROBOT PATH PLANNING BY MEANS OF CASE GRAPH AND GENETIC ALGORITHMS

P. Krček*, J. Dvořák*, J. Hodál*

Summary: *The aim of the robot path planning is searching for a path from a start to a goal position without collisions with known obstacles minimizing length, difficulty and risk of the path. A two-dimensional grid with known static obstacles models the robot environment. When planning a path, the most similar cases (already used paths or their parts) are searched for to be subsequently adapted to the new problem. A genetic algorithm or Dijkstra's algorithm is used to find the missing parts of the constructed paths or new paths if similar cases are not found or adapted solutions are not good enough.*

1. Úvod

Navigace robotu v částečně známém prostředí se často dělí na globální navigaci (*plánování cesty*) a na lokální navigaci (*plánování dráhy*). Výsledkem plánování cesty je cesta z počáteční do koncové pozice bez kolize se známými překážkami. Plánování dráhy má těsnější vazbu na akční členy robotu, protože řídí jeho pohyb podle plánu z globální navigace. Tento plán je v případě možnosti kolize s dosud neznámou překážkou lokální navigací vhodně upraven. Prostředí robotu může být modelováno dvourozměrnou mřížkou nebo spojitým prostorem. V tomto článku je zkoumáno plánování cesty autonomního mobilního robotu ve dvourozměrné mřížce.

Pokud je prostředí robotu dynamické anebo pouze částečně známé, je potom navigace robotu komplikovanou úlohou, neboť změny v prostředí je obvykle velmi obtížné modelovat. Proto je zapotřebí vyvíjet roboty se schopností učit se a zkoumat okolní prostředí. Tato schopnost je často zajišťována použitím neuronových sítí, posilovaného učení či evolučních algoritmů. Strojové učení může být také realizováno použitím případového usuzování. Případové usuzování řeší nový problém adaptací známých řešení podobných problémů, které byly již řešeny v minulosti. Zdá se, že případové usuzování je pro navigaci robotu vhodnou metodou, neboť v řadě aplikací robot řeší často podobné úkoly. Hlavní cyklus případového usuzování může být popsán čtyřmi základními kroky (Aamodt & Plaza, 1994): (i) Nalezení nejvíce podobného případu či případů; (ii) Použití informací a znalostí z tohoto případu (případů) k řešení daného problému; (iii) Kontrola navrženého řešení; (iv) Uložení vhodných částí tohoto řešení k opětnému použití při řešení budoucích podobných problémů.

* Ing. Petr Krček, RNDr. Jiří Dvořák, CSc., Ing. Jaroslav Hodál: Ústav automatizace a informatiky, Fakulta strojního inženýrství, Vysoké učení technické v Brně; Technická 2; 616 69 Brno; tel.: +420 541 142 435; e-mail: petr.krcek@seznam.cz

Případové usuzování při plánování cest robotu je nemožné použít bez některé další metody hledání cest. Přehled metod plánování cesty mobilního robotu je možno najít např. v práci Meyer & Filliat (2003). Použití těchto metod je nezbytné v situacích, kdy systém případového usuzování začíná pracovat s prázdnou bází případů, nebo když získané řešení není dost dobré a musí být přepracováno. V pracích Kruusmaa & Svensson (1998) je uvažována kombinace případového usuzování s pravděpodobnostním prohledávacím algoritmem pro globální navigaci robotu v mřížce. V tomto článku je uvažována kombinace případového usuzování s genetickými algoritmy a Dijkstrovým algoritmem. Výhoda přístupu založeného na genetických algoritmech spočívá ve schopnosti adaptovat se na změny prostředí. Genetické algoritmy pro adaptivní navigaci v prostředí reprezentovaném mřížkou jsou navrženy např. v pracích Nearchou (1999), Sugihara & Smith (1999) a Gemeinder & Gerke (2003). Dijkstrův algoritmus je používán jednak pro hledání cesty v případovém grafu, jednak v mřížce v případě selhání genetického algoritmu.

2. Okolní prostředí robotu

Uvažujeme pravoúhlou mřížku [1, m] × [1, n]. Buňka c této mřížky je určena dvojicí souřadnic: $c = (x, y)$, kde $x \in \{1, 2, \dots, m\}$, $y \in \{1, 2, \dots, n\}$. Pro jednoduchost uvažujeme čtvercový tvar buněk o velikosti větší než je velikost robotu. Předpokládáme dva typy překážek: pevné překážky, přes které není možné vést cestu, a nebezpečné překážky, přes které cesta může být vedena, ale bude ohodnocena úměrně vyšší cenou. Dále také předpokládáme, že se robot pohybuje konstantní rychlostí jen ve vodorovném, svislém a úhlopříčném směru. Tyto pohyby jsou kódované čísly $w \in \{0, 1, \dots, 7\}$.

Vzdálenost (nezohledňující překážky) mezi body $c_i = (x_i, y_i)$ a $c_j = (x_j, y_j)$ může být definována jedním z následujících vztahů:

$$d(c_i, c_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{1}$$

$$d(c_i, c_j) = |x_i - x_j| + |y_i - y_j| \tag{2}$$

$$d(c_i, c_j) = \max\{|x_i - x_j|, |y_i - y_j|\} \tag{3}$$

$$d(c_i, c_j) = \min\{|x_i - x_j|, |y_i - y_j|\}(\sqrt{2} - 1) + \max\{|x_i - x_j|, |y_i - y_j|\} \tag{4}$$

Vzdálenost definovaná vztahem (4) odpovídá tomu, že je povolen pouze vodorovný, svislý a úhlopříčný směr pohybu.

3. Případový graf

Cesta $P(c_1, c_2)$ ze startovní buňky c_1 do cílové buňky c_2 je definována jako trojice (c_1, c_2, W), kde $W = \{w_1, w_2, \dots\}$ a $w_i \in \{0, 1, \dots, 7\}$. Číslice w_i kóduje směr pohybu robotu z aktuální buňky do sousední. Tato reprezentace byla vybrána z důvodu kompatibility s použitým genetickým algoritmem.

Každá cesta v bázi případů je uložena s hodnotou funkce $F(P) = f(l, r)$, která charakterizuje cenové ohodnocení daného případu. Parametr l je délka cesty. Parametr r

charakterizuje nebezpečnost (nebo obtížnost) cesty a měl by být upravován během každého průchodu touto cestou.

Vyhledávání podobných cest je založeno na okolí buňky. Definujme *okolí* $N(c_i, \delta)$ buňky c_i jako:

$$N(c_i, \delta) = \left\{ c_j \mid d(c_i, c_j) \leq \delta \right\} \tag{5}$$

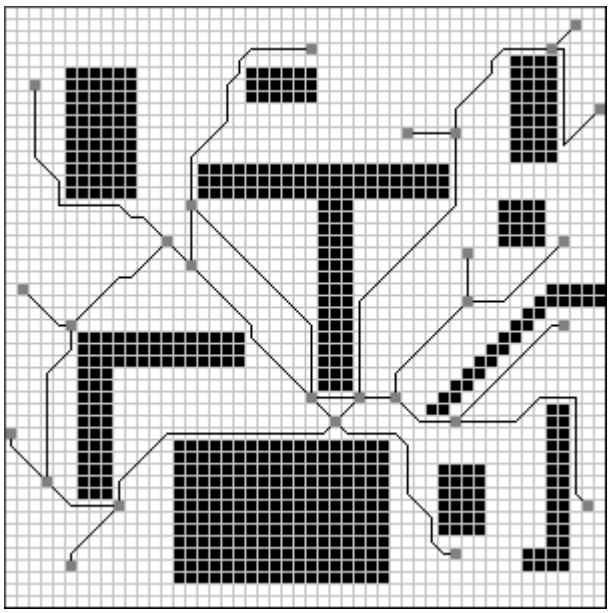
kde $d(c_i, c_j)$ je R^2_∞ –vzdálenost (3).

Pokud pro danou počáteční buňku c_s a danou cílovou buňku c_g báze případů neobsahuje cestu vedoucí z c_s do c_g, vyhledává se potom cesta podobná. V práci Kruusmaa & Svensson (1998) je podobná cesta určena vztahem:

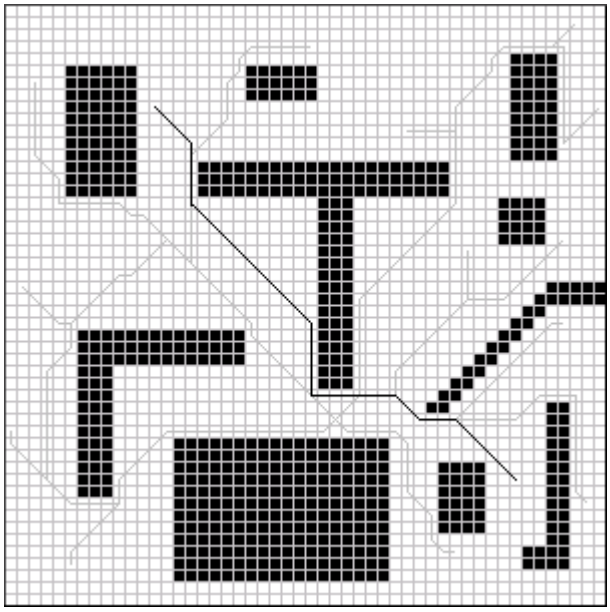
$$P(c'_s, c'_g) = \arg\min \left\{ F\big(P(c_1, c_2)\big) \,\middle|\, c_1 \in N(c_s, \delta) \wedge c_2 \in N(c_g, \delta) \right\} \tag{6}$$

To znamená, že se podobná cesta vyhledává jen z uložených kompletních cest a proto opětovné použití uložených případů je značně omezené.

Tyto nevýhody mohou být odstraněny prostřednictvím struktury zvané případový graf (*case graph*), která byla použita v práci Haigh & Shewchuk (1994) pro spojité prostředí. Při ukládání cesty do báze případů je cesta uložena buďto celá jako jeden případ, nebo je rozdělena do několika částí, které jsou uloženy jako samostatné případy (pokud se ovšem již tyto případy nebo případy jim podobné v bázi případů nevyskytují). Segmenty cest (uložené případy) mohou mít společné pouze své krajní body. Pokud cesta protíná nějaký již existující případ, pak tato cesta i případ jsou rozděleny do menších případů tak, aby byla splněna výše uvedená podmínka. Případy (segmenty) reprezentují hrany výsledného grafu a uzly tohoto grafu jsou krajními body těchto segmentů (obr. 1).



Obr. 1. Případový graf



Obr. 2. Nově nalezená cesta pomocí případů z obr. 1

Námi navrhovaný algoritmus pracující nad případovým grafem lze popsat následujícími kroky (c_s je počáteční buňka, c_g je cílová buňka):

1. Jestliže $c_s \in N(c_g, \delta)$, najdeme cestu z c_s do c_g prostřednictvím genetického algoritmu a algoritmus ukončíme.
2. Určíme množinu případů, které protínají okolí buňky c_s a množinu případů protínajících okolí buňky c_g:

$$E_s = \{C \mid C \cap N(c_s, \delta) \neq \varnothing\},\ E_g = \{C \mid C \cap N(c_g, \delta) \neq \varnothing\} \qquad (7)$$

3. Pokud jsou obě množiny E_s a E_g neprázdné, dočasně modifikujeme případový graf G takto: rozšíříme tento graf o uzel c_s a přidáme hrany spojující tuto buňku s nejbližšími buňkami případů obsažených v E_s. Tyto hrany jsou ohodnoceny odpovídajícími vzdálenostmi (bez ohledu na překážky). Stejné rozšíření provedeme pro cílovou buňku c_g. V modifikovaném grafu G' hledáme optimální cestu spojující buňky c_s a c_g. Nechť $P^*(c_1, c_2)$ je maximální segment této nalezené cesty, který se skládá jen z existujících případů nebo jejich částí. Potom dohledáme cesty $P_1(c_s, c_1)$ a $P_2(c_2, c_g)$ pomocí genetického algoritmu. Hledaná cesta vznikne spojením těchto tří segmentů: $P_1(c_s, c_1)$, $P^*(c_1, c_2)$ a $P_2(c_2, c_g)$. Algoritmus ukončíme.
4. Pokud optimální cesta není nalezena (graf G' není souvislý), určíme komponenty G'_s a G'_g obsahující buňky c_s a c_g. Pak přidáme do G' hrany spojující dvojice nejbližších uzlů z G'_s a z G'_g, které ohodnotíme vzdálenostmi těchto uzlů. V grafu G' nyní hledáme optimální cestu spojující buňky c_s a c_g. Nechť $P_s(c_1^s, c_2^s)$ a $P_g(c_1^g, c_2^g)$ jsou maximální

segmenty těchto nalezených cest, které se skládají jen z existujících případů nebo jejich částí ležících v G'_s a G'_g. Nyní dohledáme cesty $P_1(c_s, c_1^s)$, $P_2(c_2^s, c_1^g)$ a $P_3(c_2^g, c_g)$ prostřednictvím genetického algoritmu. Hledaná cesta vznikne spojením těchto pěti segmentů: $P_1(c_s, c_1^s)$, $P_s(c_1^s, c_2^s)$, $P_2(c_2^s, c_1^g)$, $P_g(c_1^g, c_2^g)$ a $P_3(c_2^g, c_g)$. Algoritmus ukončíme.

5. Jestliže je neprázdná jen jedna z množin E_s a E_g, např. E_s, potom dočasně modifikujeme graf G tak, že do něj přidáme uzel c_s a hrany spojující tuto buňku s nejbližšími buňkami případů obsažených v E_s ohodnocené příslušnými vzdálenostmi. Určíme komponentu G'_s obsahující buňku c_s a do G' přidáme hrany spojující uzel c_g s nejbližšími uzly z G'_s, které ohodnotíme příslušnou vzdáleností. V grafu G' nyní hledáme optimální cestu spojující buňky c_s a c_g. Nechť $P_s(c_1^s, c_2^s)$ je maximální segment této nalezené cesty, který se skládá jen z existujících případů nebo jejich částí. Potom dohledáme cesty $P_1(c_s, c_1^s)$ a $P_2(c_2^s, c_g)$ prostřednictvím genetického algoritmu. Hledanou cestu získáme spojením segmentů $P_1(c_s, c_1^s)$, $P_s(c_1^s, c_2^s)$, $P_2(c_2^s, c_g)$ a algoritmus ukončíme. Pokud je neprázdná pouze množina E_g, postupujeme obdobně.

6. Pokud jsou obě množiny E_s i E_g prázdné, cestu z c_s do c_g najdeme pomocí genetického algoritmu a algoritmus ukončíme.

Nechť P je cesta absolvovaná robotem. Tato cesta může být odlišná od cesty nalezené výše popsaným algoritmem, protože systém lokální navigace ji může v případě výskytu neznámé nebo dynamické překážky změnit. Z cesty P uvažujeme pro uchování v bázi případů jen ty segmenty, které nejsou sestaveny z existujících případů (tj. segmenty nalezené genetickým algoritmem nebo systémem lokální navigace). Segment S je uchován pouze tehdy, pokud báze případů neobsahuje žádný případ, který je podobný S a má nižší ohodnocení. Podobné případy s vyšším ohodnocením budou segmentem S nahrazeny, ovšem tak, aby nedošlo ke zvýšení počtu komponent souvislosti případového grafu. Případ C považujeme za podobný případu S, jestliže platí

$$d(c_1^C, c_1^S) \le \varepsilon \wedge d(c_2^C, c_2^S) \le \varepsilon \vee d(c_1^C, c_2^S) \le \varepsilon \wedge d(c_2^C, c_1^S) \le \varepsilon \qquad (8)$$

kde c_1^C, c_2^C jsou krajní body případu C a c_1^S, c_2^S jsou krajní body případu S.

4. Genetický algoritmus

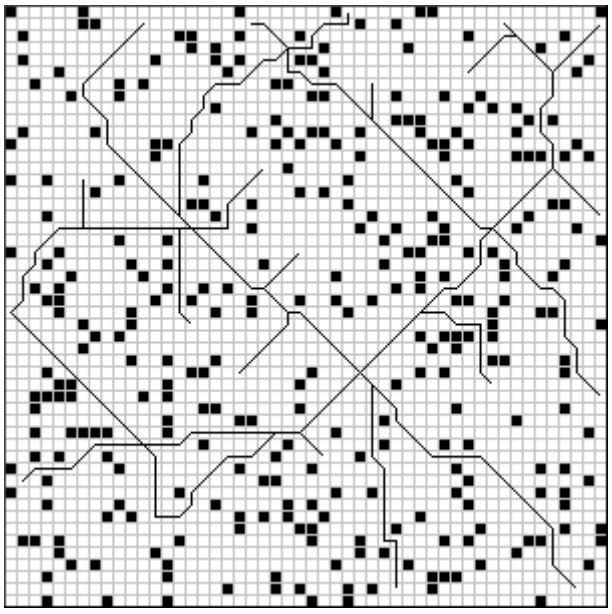
Pro jednoduchost uvažujeme pro plánování cesty genetický algoritmus, který je implementovaný v práci Sedláček (2000). Tato metoda používá chromozómy pevné délky, která závisí na velikosti prohledávaného prostoru. Chromozómy jsou posloupnosti $S = \{w_1, w_2, \ldots, w_N\}$, kde každý gen $w_i \in \{0, 1, \ldots, 7\}$ kóduje směr pohybu robotu do sousední buňky. Každý chromozóm reprezentuje cestu z dané počáteční buňky c_s, ale tato cesta nemusí obsahovat cílovou buňku c_g. Pokud je cílová buňka dosažena dříve, než jsou vyčerpány všechny geny, jsou zbývající geny ignorovány.

Počáteční populace je generována náhodně. Kvalitu chromozómů určují dvě fitness funkce aplikované v následujícím pořadí. První funkce reprezentuje vzdálenost mezi koncovou buňkou cesty a cílovou buňkou c_g, druhá funkce reprezentuje cenu této cesty. Algoritmus používá binární turnajovou selekci, uniformní křížení, mutaci jednoho náhodně generovaného genu a inkrementální výměnu populace. Výpočet algoritmu je ukončen po dosažení daného počtu generací. V kombinaci s případovým usuzováním, kde genetické algoritmy vyhledávají části výsledné cesty, je počet generací a velikost chromozomů určována v závislosti na vzdálenosti $d(c_s, c_g)$ dané vztahem (3).

V případě nenalezení cesty pomocí genetického algoritmu v daném časovém limitu se vytvoří z mřížky graf, jehož uzly tvoří všechny buňky mřížky bez pevných překážek a hrany odpovídají všem možným směrům pohybu robotu do sousedních buněk. Ohodnocení těchto hran jsou součtem vzdáleností sousedních buněk vypočtených podle (1) a z ceny těchto buněk. K vyhledání cesty v takto získaném grafu je potom použito Dijkstrova algoritmu, jenž má sice větší časovou obtížnost, ale vždy zaručuje nalezení optimální cesty.

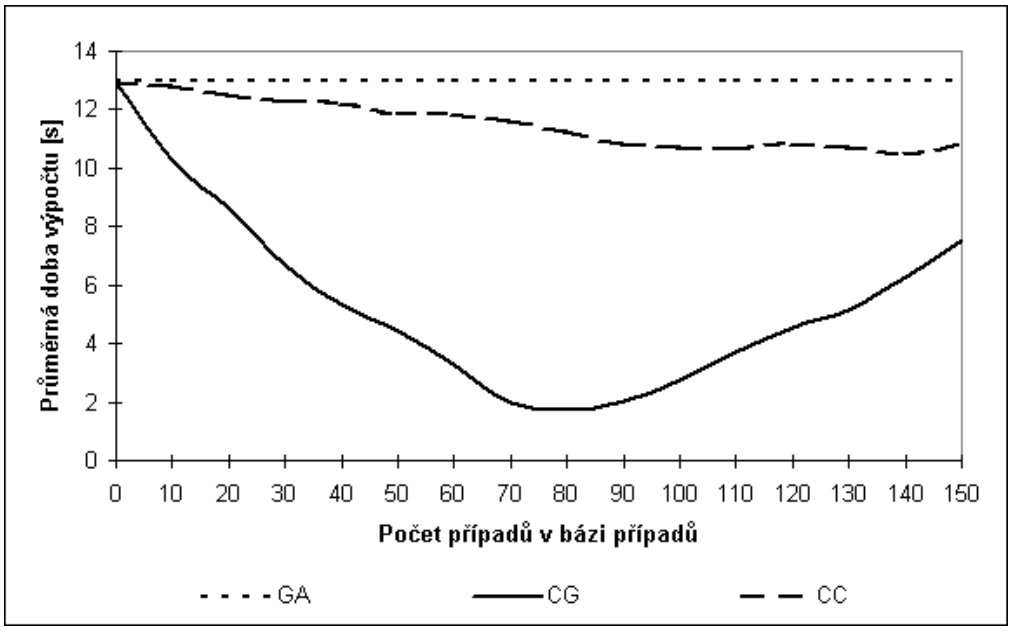
5. Simulační experimenty

Pro ověření a porovnání algoritmů bylo vytvořeno simulační prostředí. Pevné překážky byly generovány náhodně a tvořily 10% buněk (příklad překážek spolu s vytvořeným případovým grafem ukazuje obr. 3). Byly porovnány tři metody plánování cesty: metoda GA založená pouze na genetických algoritmech (s případným vyhledáváním pomocí Dijkstrova algoritmu), metoda CC kombinující použití kompletních cest jako případů (Kruusmaa & Svensson, 1998) s vyhledáváním pomocí GA a v tomto článku navržená metoda CG založená na případovém grafu a metodě GA. Genetický algoritmus byl ve všech metodách ukončen po nalezení prvního řešení. Obr. 4 ukazuje závislost průměrné doby hledání jedné cesty na počtu uložených případů pro mřížku 200 × 200. Obr. 5 zachycuje stejnou závislost jako obr. 4, ale pro menší mřížku 50 × 50.

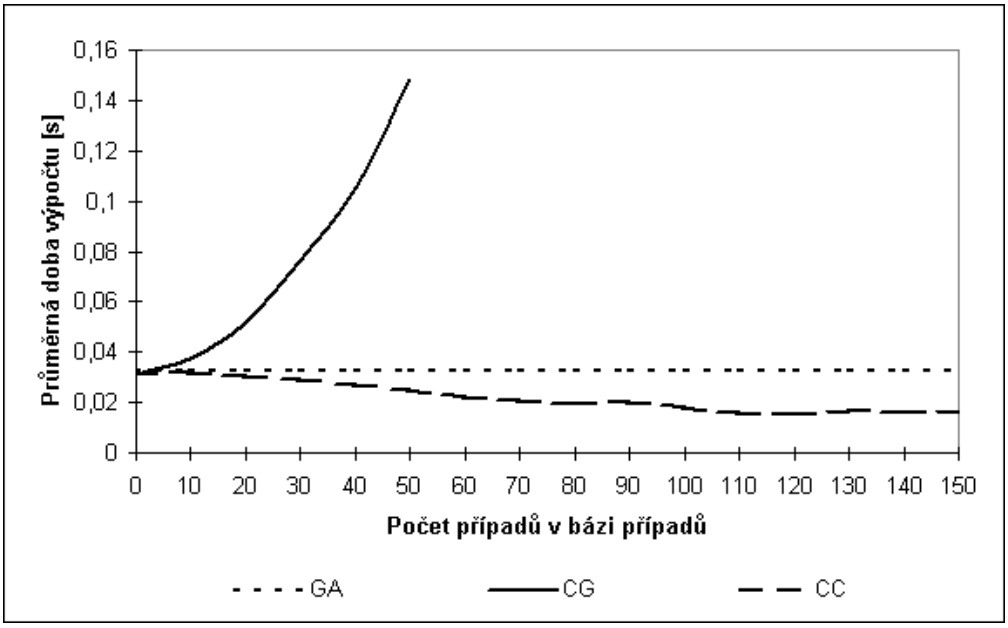


Obr. 3. Náhodné překážky v mapě a cesty v bázi případů v mřížce 50 x 50

Z provedených experimentů vyplývá, že použití metody CG je vhodné až pro větší mřížky. Pro menší mřížky je zbytečné používat jakoukoliv metodu založenou na případovém usuzování, neboť v těchto malých mřížkách je doba výpočtu pomocí klasických vyhledávacích metod přijatelná, dokonce např. použití metody CG pro malé mřížky vede ke zvětšení průměrné doby výpočtu (viz obr. 5). Naopak použití případového usuzování ve větších mřížkách vede k výraznému snížení doby výpočtu. Pro efektivní funkci metody CG je však zapotřebí, aby velikost báze případů byla udržována ve vhodných mezích (pro mřížku 200 × 200 na obr. 4 je to přibližně 70 – 90 případů).



Obr. 4. Závislost průměrné doby hledání na počtu případů v mřížce 200 × 200



Obr. 5. Závislost průměrné doby hledání na počtu případů v mřížce 50 × 50

6. Závěr

Tento článek se zabývá plánováním cesty autonomního mobilního robotu ve dvourozměrné pravoúhlé mřížce, jejíž buňky mohou obsahovat známé překážky. Navrhujeme zde metodu kombinující případové usuzování s hledáním pomocí genetických algoritmů. Tato metoda je založena na struktuře zvané případový graf, která je sestavena z částí již dříve nalezených cest. Z dosud provedených experimentů vyplývá, že použití případového grafu pro velké mřížky přináší významnou úsporu ve srovnání se samotným genetickým algoritmem a s použitím celých cest jako případů. Je ovšem nutné udržovat velikost báze případů ve vhodných mezích, které závisejí mimo jiné na velikosti mřížky.

V dalším výzkumu se budeme věnovat problematice související s uchováváním případů a udržováním báze případů. Také budeme analyzovat dopady jiných reprezentací cest v chromozómech a možnosti kombinace případového usuzování s jinými metodami.

7. Poděkování

Tento článek byl zpracován v rámci vědecko-výzkumného záměru MSM 0021630518 "Simulační modelování mechatronických soustav".

8. Literatura

Aamodt, A. & Plaza, E. (1994) Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. *AI Communications*, No. 5, v. 7, pp. 39-59.

Gemeinder, M. & Gerke, M. (2003) GA-based path planning for mobile robot systems employing an active search algorithm. *Applied Soft Computing*, v. 3, pp. 149-158.

Haigh, K. & Shewchuk, J. (1994) Geometric Similarity Metrics for Case-Based Reasoning, in: *Case-Based Reasoning: Working Notes from the AAAI-94 Workshop*, Seattle, WA, August, AAAI Press, pp. 182-187.

Kruusmaa, M. & Svensson, B. (1998) Combined Map-Based and Case-Based Path Planning for Mobile Robot Navigation, in: *Proceedings of International Symposium of Intelligent Robotic Systems*, January 10-12, 6 pp.

Kruusmaa, M. & Svensson, B. (1998) Using Case-Based Reasoning for Mobile Robot Path Planning, in: *Proceedings of the 6th German Workshop on Case-Based Reasoning*, March 6-8, Berlin, 8 pp.

Meyer, J.-A. & Filliat, D. (2003) Map-based navigation in mobile robots: II. A review of map-learning and path-planning strategies. *Cognitive Systems Research*, v. 4, pp. 283-317.

Nearchou, A. (1999) Adaptive Navigation of Autonomous Vehicles Using Evolutionary Algorithms. *Artificial Intelligence in Engineering*, No. 2, v. 13, pp. 159-173.

Sedláček, M. (2000) *Optimalizace trajektorie robota*. Diplomová práce, FSI VUT, Brno.

Sugihara, K. & Smith, J. (1999) Genetic Algorithms for Adaptive Planning of Path and Trajectory of a Mobile Robot in 2D Terrains. *IEICE Transactions on Information and Systems*, No. 1, v. E82-D, pp. 309-317.