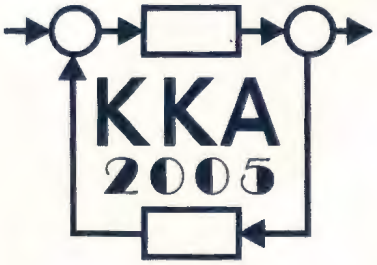
XV Krajowa Konferencja Automatyki

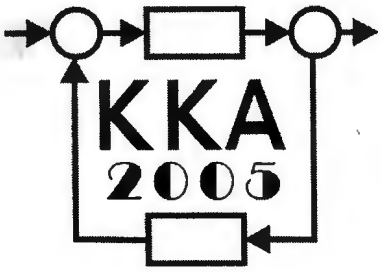
Tom II



Redaktorzy:
Zdzisław Bubnicki
Roman Kulikowski
Janusz Kacprzyk

XV Krajowa Konferencja Automatyki

Tom II



Redaktorzy:
Zdzisław BUBNICKI
Roman KULIKOWSKI
Janusz KACPRZYK

ORGANIZATOR
Komitet Automatyki i Robotyki Polskiej Akademii Nauk
Instytut Badań Systemowych Polskiej Akademii Nauk
WSPÓŁORGANIZATORZY
Politechnika Warszawska
Przemysłowy Instytut Automatyki i Pomiarów
Polskie Stowarzyszenie Pomiarów, Automatyki i Robotyki

ORGANIZATOR

Komitet Automatyki i Robotyki Polskiej Akademii Nauk
Instytut Badań Systemowych Polskiej Akademii Nauk

WSPÓŁORGANIZATORZY

Politechnika Warszawska
Przemysłowy Instytut Automatyki i Pomiarów
Polskie Stowarzyszenie Pomiarów, Automatyki i Robotyki

KOMITET PROGRAMOWY

Przewodniczący	Zdzisław BUBNICKI
Zastępca Przewodniczącego	Roman KULIKOWSKI

CZŁONKOWIE

Stanisław BAŃKA	Michał BIAŁKO
Mikołaj BUSŁOWICZ	Władysław FINDEISEN
Ryszard GESSING	Henryk GÓRECKI
Jakub GUTENBAUM	Jerzy JÓZEFCZYK
Stanisław KACZANOWSKI	Tadeusz KACZOREK
Janusz KACPRZYK	Jerzy KLAMKA
Józef KORBICZ	Zbigniew KOWALSKI
Krzysztof KOZŁOWSKI	Juliusz L. KULIKOWSKI
Krzysztof KUŹMIŃSKI	Kazimierz MALANOWSKI
Krzysztof MALINOWSKI	Wojciech MITKOWSKI
Antoni NIEDERLIŃSKI	Władysław PEŁCZEWSKI
Tadeusz PUCHAŁKA	Leszek RUTKOWSKI
Stanisław SKOCZOWSKI	Roman SŁOWIŃSKI
Jerzy ŚWIĄTEK	Andrzej ŚWIERNIAK
Ryszard TADEUSIEWICZ	Piotr TATJEWSKI
Krzysztof TCHOŃ	Leszek TRYBUS
Jan WĘGLARZ	Andrzej P. WIERZBICKI

KOMITET ORGANIZACYJNY

Przewodniczący	Roman KULIKOWSKI
Zastępcy Przewodniczącego	Janusz KACPRZYK
	Stanisław KACZANOWSKI
	Tadeusz KACZOREK
	Krzysztof MALINOWSKI
Członkowie	Roman OSTROWSKI
	Tadeusz PUCHAŁKA
	Dariusz WAGNER
Sekretarze naukowi	Jan STUDZIŃSKI
	Jan W. OWSIŃSKI

ISBN 83-89475-01-4

Druk: ARGRAF, Warszawa

STEROWANIE KOMPLEKSAMI OPERACJI

REDUKCJA SYSTEMU STRIPS Z UWZGLĘDNIENIEM NIEPEŁNOŚCI INFORMACJI DO ZADANIA PROGRAMOWANIA LINIOWEGO[†]

Adam GAŁUSZKA[*], Andrzej ŚWIERNIAK[**]

* Politechnika Śląska, Instytut Automatyki
ul. Akademicka 16, 44-100 Gliwice, e-mail: Adam.Galuszka@polsl.pl

** Politechnika Śląska, Instytut Automatyki
ul. Akademicka 16, 44-100 Gliwice, e-mail: Andrzej.Swierniak@polsl.pl

Streszczenie: W pracy sprowadzono problem planowania STRIPS w środowisku tzw. świata klocków do problemu programowania liniowego. Takie podejście pozwala efektywniej rozwiązać problemy rzeczywistych (dużych) rozmiarów, jednak przy założeniu, że otrzymane rozwiązanie posiada interpretację w świecie klocków. Ponieważ rozwiązania otrzymane z zadania programowania liniowego są wartościami z przedziału <0,1>, a wartości te odpowiadają stopniu spełnienia warunków w systemie STRIPS („0" odpowiada niespełnienie warunku, „1" odpowiada całkowite spełnienie), zaproponowana metodologia może służyć analizie problemów STRIPS z uwzględnieniem niepełności informacji.

Słowa kluczowe: Zadanie programowania liniowego, świat klocków, planowanie w warunkach niepełnej informacji.

1. WPROWADZENIE

Próby implementacji robota inteligentnego pociągają za sobą konieczność analizy i rozwiązywania takich zadań jak percepcja i rozumienie otoczenia, osiąganie celu, planowanie zadań, wykonywanie zadań i monitorowanie ruchów robota. Przez robot inteligentny rozumiemy raczej robot autonomiczny, zdolny do analizy otoczenia, planowania i wykonywania ruchów w otoczeniu niezdeterminowanym [16], [18]. Jądrem implementacji takiego robota jest system, w którym zapisana wiedza oraz określona metoda poszukiwania rozwiązań problemów pozwala na generowanie planu ruchów robota.

W ogólnym przypadku powyższy problem jest bardzo złożony. Wynika to ze złożoności i sposobu monitorowania otoczenia robota, liczby stopni swobody robota wynikających z jego konstrukcji mechanicznej, liczby poziomów sterowania robotem, złożoności inteligentnego oprogramowania. Wszystko to spowodowało, że inteligentna robotyka stała się dziedziną interdyscyplinarną [16]. Praktyczne implementacje inteligentnych systemów wskazują, że zarówno procesy planowania zadań jak i planowania trajektorii powinny uwzględniać niepewność dostępnej informacji [18].

[†]Praca była finansowana z projektu badawczego KBN w roku 2005.

Obecnie problemy poprawy efektywności planowania z wykorzystaniem reprezentacji STRIPS oraz planowania w warunkach niepewności znajdują się w centrum zainteresowań Sztucznej Inteligencji [1], [17], [24], [25], [19], [11], [12], [13], [14], [22], [23]. W niniejszej pracy rozważane są problemy planowania zadań i osiągania celów w środowisku tzw. świata klocków. W szczególności analizowane są: sposób redukcji złożoności obliczeniowej oraz modelowanie planowania jako systemu zawierającego niepewną (niejednoznaczną) informację o stanie początkowym problemu. Tego typu problemy mogą być reprezentowane za pomocą systemu STRIPS [6].

W pracy sprowadzono problem planowania o reprezentacji STRIPS modelujący środowisko tzw. świata klocków do zadania programowania liniowego (ZPL). Takie podejście może redukować klasę złożoności obliczeniowej problemu planowania. Dzieje się tak dlatego, że:

- problem znalezienia planu optymalnego (o najmniejszej liczbie operatorów) w świecie klocków jest NP zupełny [8];
- transformacja do zadania programowania liniowego jest wielomianowa (tzn. czas transformacji jest ograniczony wielomianem zależnym od rozmiaru problemu) [3];
- zadanie programowania liniowego jest problemem wielomianowo złożonym (należy do klasy P złożoności obliczeniowej) [4], [9].

Ponieważ rozwiązania otrzymane z ZPL (zadania programowania liniowego) są wartościami z przedziału <0,1>, a wartości te odpowiadają stopniu spełnienia warunków w systemie STRIPS („0" odpowiada niespełnienie warunku, „1" odpowiada całkowite spełnienie), zaproponowana metodologia może służyć analizie problemów STRIPS z uwzględnieniem niepewności. Niepewność w problemach planowania typu STRIPS jest często reprezentowana jako niejednoznaczność sytuacji początkowej problemu (tzn. zakładamy, że mamy do

czynienia z alternatywą możliwych sytuacji początkowych – [22], [25]), wynikająca z niepełności dostępnej informacji. Takie założenie uczyniono w pracy. Koszt uwzględnienia niepełności informacji (czyli zbliżenia się w modelowaniu do sytuacji rzeczywistej) to wzrost złożoności obliczeniowej problemu, który teraz w przypadku ogólnym NP^{NP} -zupełny ($\Sigma_2 P$ – zupełny) [1].

Jako współczesne zastosowania problemów typu „świat klocków" posiadających reprezentację STRIPS można wymienić:

- środowisko testowe (benchmark) dla problemów planowania (odpowiednik problemu komiwojażera dla algorytmów przeszukiwania grafu) [10];
- reprezentacja dla problemów logistycznych (magazynowania, planowania dostaw - przesuwanie klocków to przesuwanie pakunków, ciężarówek, samolotów itp.) [20];
- reprezentacja dla problemów załadunku (dla świata klocków z ograniczonym miejscem na stole) [21].

Wynikiem pracy jest sprowadzenie problemu planowania STRIPS w środowisku tzw. świata klocków do problemu programowania liniowego i analiza wpływu niepełności informacji na rozwiązanie zadania liniowego. Pokazano również zależność rozmiaru problemu liniowego od rozmiaru świata klocków.

2. PROBLEM PLANOWANIA TYPU STRIPS

Problem planowania STRIPS (*propositional STRIPS planning problem*) składa się z czwórki $<P, O, I, G>$ ([2], [15]), gdzie :

- P jest skończonym zbiorem podstawowych formuł, zwanych warunkami (*conditions*),
- jest skończonym zbiorem operatorów, gdzie każdy operator $o \in O$ przyjmuje formę $o^+, o^- \Rightarrow o_+, o_-$, gdzie
- $o^+ \subseteq P$ są pozytywnymi warunkami stosowalności operatora (*positive preconditions*),
- $o^- \subseteq P$ są negatywnymi warunkami stosowalności operatora (*negative preconditions*),
- $o_+ \subseteq P$ są pozytywnymi skutkami zastosowania operatora (*positive postconditions*),
- $o_- \subseteq P$ są negatywnymi skutkami zastosowania operatora (*negative postconditions*),
- $I \subseteq P$ jest stanem początkowym zadania (*initial state*),
- $G = <G_+, G_->$ jest spełnialną koniunkcją pozytywnych i negatywnych warunków, w której $G_+ \subseteq P$ są pozytywnymi warunkami a $G_- \subseteq P$ są negatywnymi warunkami.

Każdy stan problemu planowania może być opisany jako podzbiór $S \subseteq P$, w taki sposób, że warunek $p \in P$ jest spełniony (prawdziwy) w opisie stanu, jeśli $p \in S$ i niespełniony (fałszywy) w przeciwnym wypadku. Zbiór I precyzuje, które warunki są prawdziwe a które fałszywe w stanie początkowym problemu, tzn. warunek $p \in P$ jest prawdziwy w stanie początkowym jeśli $p \in I$ lub fałszywy w przeciwnym przypadku. Zbiór G precyzuje cel, tzn. $S \subseteq P$ jest stanem docelowym problemu jeśli S spełnia G, tzn. jeśli $G_+ \subseteq S$ i $G_- \cap S = \varnothing$.

Efekt zastosowania skończonej sekwencji operatorów do stanu S może być opisany poprzez funkcje *Result*, która jest zdefiniowana następująco (symbol "\" oznacza różnice zbiorów):

$Result(S, <>) = S$,

$Result(S, <o>) = (S \cup o_+) \setminus o_-$, jeśli $o^+ \subseteq S \wedge o^- \cap S = \varnothing$;
S, w przeciwnym przypadku,

$Result(S, <o_1, o_2, \ldots, o_n>) =$
$= Result(Result(S, <o_1>), <o_2, \ldots, o_n>)$.

Z definicji funkcji *Result* wynika, że każdy operator może być zastosowany do każdego stanu, natomiast tylko ten operator daje efekt, dla którego spełnione są warunki stosowalności (*positive and negative preconditions*). Dany operator może występować wiele razy w sekwencji operatorów tworzących plan. Skończona sekwencja operatorów $<o_1, o_2, \ldots, o_n>$ jest rozwiązaniem problemu planowania, jeśli stan wynikający z zastosowania funkcji $Result(I, <o_1, o_2, \ldots, o_n>)$ jest stanem docelowym problemu.

3. TRANSFORMACJA DO ZADANIA PROGRAMOWANIA LINIOWEGO

Idea transformacji do ZPL polega na przypisaniu każdemu warunkowi i operatorowi systemu STRIPS w każdym etapie planowania zmiennej [3]. Zakładamy, że jeśli p jest warunkiem i jeśli planowanie jest podzielone na l etapów, to potrzebujemy $l+1$ zmiennych dla tego warunku: $p(0), p(1), \ldots, p(l)$. Jeśli z kolei op będzie operatorem, to potrzebujemy l zmiennych dla każdego operatora: $op(0), op(1), \ldots, op(l-1)$. Funkcja celu osiąga wartość maksymalną, jeśli wartości warunków definiujących cel w ostatnim etapie planowania są równe 1.

Rozważmy przykład ze środowiska świata klocków [7]. Ograniczmy problem jedynie do dekompozycji stanu początkowego pewnej konfiguracji klocków. Dla prostoty założono następującą prostą konfigurację klocków:

on(A,B) ^ on(D,C)

dysponujemy jednym operatorem umożliwiającym dekompozycję:

move-to-table(x,y) – oznacza ściągnięcie klocka x z y i położenie na stole

preconditions: on(x,y), clear(x)

postconditions: not(on(x,y)), clear(y)

Stan będzie zdekomponowany, jeśli dla każdego klocka będzie prawdziwy warunek:

clear(A), clear(B), clear(C), clear(D)

Załóżmy 2 etapy planowania (dla każdej z możliwych założonych sytuacji początkowych wystarczą 2 operatory do ich dekompozycji). Wtedy system STRIPS dla problemu dekompozycji jest zdefiniowany poniżej:

$$P = \{on(x,y), clear(x)\}$$
$$O = \{move-to-table(x,y) : on(x,y) \wedge clear(x) \Rightarrow \neg on(x,y) \wedge clear(y)\}$$
$$I = \{on(A,B), on(D,C), clear(A), clear(D)\}$$
$$G = \{clear(A) \wedge clear(B) \wedge clear(C) \wedge clear(D)\}$$

Tak więc mamy (16+16+16) zmiennych dla warunków:

on(A,B)(i)	*on(B,C)(i)*	*on(C,D)(i)*	*clear(A)(i)*
on(A,C)(i)	*on(B,D)(i)*	*on(D,A)(i)*	*clear(B)(i)*
on(A,D)(i)	*on(C,A)(i)*	*on(D,B)(i)*	*clear(C)(i)*
on(B,A)(i)	*on(C,B)(i)*	*on(D,C)(i)*	*clear(D)(i)*

$i = 0, 1, 2$.
Dodatkowo potrzebujemy 24 zmienne dla operatorów (12+12):

move-to-table(A,B)(i)	*move-to-table(C,A)(i)*
move-to-table(A,C)(i)	*move-to-table(C,B)(i)*
move-to-table(A,D)(i)	*move-to-table(C,D)(i)*
move-to-table(B,A)(i)	*move-to-table(D,A)(i)*
move-to-table(B,C)(i)	*move-to-table(D,B)(i)*
move-to-table(B,D)(i)	*move-to-table(D,C)(i)*

Funkcja celu definiuje stan docelowy:
Max (clear(A)(2) + clear(B)(2) + clear(C)(2) + clear(D)(2))

Dla rozwiązanego problemu planowania funkcja celu będzie równa 4 (4 warunki będą prawdziwe).
Co najwyżej 1 operator może być zastosowany na każdym etapie:
$\Sigma move\text{-}to\text{-}table(x,y)(1) \leq 1$
$\Sigma move\text{-}to\text{-}table(x,y)(2) \leq 1$
Operator nie może być zastosowany, jeśli jego warunki stosowalności nie są prawdziwe:
$on(x,y)(i) \geq move\text{-}to\text{-}table(x,y)(i)$ - dla wszystkich 12 operatorów w i-tym etapie;
$clear(A)(i) \geq move\text{-}to\text{-}table(A,B)(i) + move\text{-}to\text{-}table(A,C)(i) + move\text{-}to\text{-}table(A,D)(i)$
$clear(B)(i) \geq move\text{-}to\text{-}table(B,A)(i) + move\text{-}to\text{-}table(B,C)(i) + move\text{-}to\text{-}table(B,D)(i)$
$clear(C)(i) \geq move\text{-}to\text{-}table(C,A)(i) + move\text{-}to\text{-}table(C,B)(i) + move\text{-}to\text{-}table(C,D)(i)$
$clear(D)(i) \geq move\text{-}to\text{-}table(D,A)(i) + move\text{-}to\text{-}table(D,B)(i) + move\text{-}to\text{-}table(D,C)(i)$
$i = 0, 1$.
Następna grupa ograniczeń opisuje zmiany wartości zmiennych dla warunków w następnych etapach po zastosowaniu operatorów. Są to ograniczenia równościowe:
$clear(A)(i+1) = clear(A)(i) + move\text{-}to\text{-}table(B,A)(i) + move\text{-}to\text{-}table(C,A)(i) + move\text{-}to\text{-}table(D,A)(i)$
i tak dla wszystkich pozostałych warunków: *clear(x)(i+1)*. Potrzebne są jeszcze równania opisujące zmiany warunków typu: on(x,y):
$on(A,B)(i+1) = on(A,B)(i) - move\text{-}to\text{-}table(A,B)(i)$
Wreszcie ograniczenia na wartości zmiennych: wartości te muszą zawierać się w przedziale <0,1> odpowiadającym wartościom prawdy dla danych warunków.

Otrzymany w wyniku powyższej transformacji problem ZPL ma rozmiar (dla „n" klocków i „l" kroków planowania, gdzie zawsze $l \leq 2n$):
- liczba zmiennych: mniej niż $2l(n^2-1)$;
- liczba ograniczeń nierównościowych: $l\,(n^2 + n + 1)$;
- liczba ograniczeń równościowych: $l\,(n^2 + n)$.

W wyniku transformacji (wielomianowej), otrzymujemy ZPL rozmiarów sprawiających problemy dzisiejszym metodom obliczeniowym, np. dla zaledwie 30 klocków i 15 kroków rozwiązujących problem ZPL ma rozmiary już:
- liczba zmiennych: 28 380
- liczba ograniczeń nierównościowych: 13 965
- liczba ograniczeń równościowych: 13 950.

Liczba stanów problemu świata 30 klocków to [20]:
1979874012955717189150059823979685 1.

4. PRZYKŁADY

Dla ilustracji zagadnienia transformacji zaimplementowano w środowisku MATLAB 2 przykłady. W pierwszym przypadku był to problem z kompletną informacją o transformowanym środowisku, w przypadku drugim – z informacją niekompletną. W obydwu przypadkach po transformacji do ZPL otrzymujemy problem o 72 zmiennych, 34 ograniczeniach nierównościowych i 32 równościowych. Problemy większych rozmiarów (model i wyniki symulacji):
www.zts.ia.polsl.gliwice.pl/galuszka/blocks_LP.htm .

Problem pierwszy do dekompozycja świata klocków o sytuacji początkowej danej jako zbiór:
on(A,B), on(D,C), clear(A), clear(D)
Rozwiązaniem ZPL jest zbiór wartości zmiennych, które dają optymalną wartość funkcji celu. Wartości te korespondują z wartościami prawdy warunków i operatorów:
move-to-table(D,C)(0)=1 i *move-to-table(A,B)(1)=1*
które rozwiązują problem dekompozycji w 2 krokach.

Przypadek drugi reprezentuje problem z niejednoznaczną informacją o stanie początkowym. Teraz sytuacja początkowa składa się z dwóch możliwych stanów początkowych:
(on(A,B), on(D,C), clear(A), clear(D)) or (on(A,C), on(D,B), clear(A), clear(D))

W ZPL wartości zmiennych odpowiadające predykatom:
on(A,B)(0), on(D,C)(0), on(A,C)(0), on(D,B)(0)
są równe 0.5, co reprezentuje przypadek niejednoznaczności następującej sytuacji: czy blok *A* jest na *C* lub *B*, a blok *D* jest na *C* lub *B*.

Rozwiązanie ZPL w przypadku drugim daje wartości prawdy dla operatorów:
move-to-table(A,B)(0)=0.5
move-to-table(A,C)(0)=0.5
move-to-table(D,B)(1)=0.5
move-to-table(D,C)(1)=0.5
które dają rozwiązanie problemu również w dwóch krokach. Interpretacja wartości prawdy jest następująca: w kroku pierwszym ściągnąć blok A na stół (z bloku B lub C), następnie ściągnąć blok D z B lub C.

Powyższe przykłady pokazują przydatność transformacji problemów planowania w przypadku problemów z pewną i niepewną informacją. Nie wiadomo jednak, jak duża jest grupa problemów planowania, dla których rozwiązanie problemu po transformacji daje rezultaty, które można poprawnie zinterpretować w danej dziedzinie (tutaj – świat klocków). Określenie tej grupy będzie tematem dalszych prac.

5. PODSUMOWANIE I WNIOSKI

Redukcja do ZPL redukuje klasę złożoności obliczeniowej problemu planowania przy założeniu, że otrzy-

mane rozwiązanie posiada interpretację w świecie klocków.
Programowanie całkowitoliczbowe da w wyniku wartości zmiennych 0 lub 1, co odpowiadałoby rozwiązaniu problemu planowania bez uwzględniania niepewności, co do warunków w stanie początkowym problemu. Podejście takie jest jednak nieefektywne, gdyż problem programowania całkowitoliczbowego jest NP-zupełny, w wyniku czego redukowalibyśmy problem NP-zupełny do innego NP-zupełnego, co nie jest postępem z punktu widzenia zwiększania efektywności obliczeniowej. Programowanie liniowe da w wyniku wartości z przedziału <0,1> włącznie, co pozwala uwzględniać niepewność, co do stopnia prawdy warunków w stanie początkowym problemu, jednak nie zawsze otrzymany wynik daje poprawną interpretację znalezionego planu. Jest to koszt, jaki ponosimy redukując problem planowania do wielomianowego.

REDUCTION OF BLOCK WORLD PROBLEM IN THE PRESENCE OF INCOMPLEETENESS TO LINEAR PROGRAMMING

Abstract: Planning with complete information is usually at least NP-complete problem (e.g. optimal planning in Block World environment is NP-complete). Planning in the presence of incompleteness belongs to the next level in the hierarchy of completeness. It is assumed that incompleteness is when in the problem there are some possible initial situations and one goal state. To reduce computational complexity of this approach a transformation to Linear Programming problem is proposed. The transformation from planning to Linear Programming is based on mapping of conditions and operators in each plan step to variables. Truth-values of conditions are mapped to 0 and 1 for the planning without incompleteness, and to any values between 0 and 1 for planning with incomplete information. The objective function reaches the maximum if the goal situation is true in last step of planning. Examples illustrate the reduced problem.

Literatura

[1] Baral Ch., Kreinovich V., Trejo R. (2000) Computational complexity of planning and approximate planning in the presence of incompleteness. *Artificial Intelligence*, 122, 241-267.
[2] Bylander T. (1994) The computational complexity of propositional STRIPS planning. *Artificial Intelligence*, 69,165-204.
[3] Bylander T. (1997) A Linear Programming Heuristic for Optimal Planning. *Int. Conf. American Association for Artificial Intelligence*, 1997, www.aaai.org.
[4] Cormen T.H., Ch.E. Leiserson, R.L. Rivest. (1994) *Introduction to Algorithms*. The Massachusetts Institute of Technology.
[5] Dougherty, E.R., Giardina Ch.R. (1988) *Mathematical Methods for Artificial Intelligence and Autonomous Systems*. Prentice-Hall International, Inc. USA.
[6] Gałuszka A. (1999) „Distinguish- and indistinguishability of STRIPS planning problem elements". *In Proceedings of the 11th European Simulation Symposium,* Erlangen, 95-99.
[7] Gałuszka A., Świerniak A. (2004) Translation STRIPS Planning in Multi-Robot Environment to Linear Programming. *Lecture Notes in Artificial Intelligence*, **3070**, 768-773.
[8] Gupta N., Nau D.S. (1992) On the complexity of Blocks-World Planning. *Artificial Intelligence*, **56** 2/3, 223-254.
[9] Garey M.R. Johnson D.S. (1979) *Computers and Intractability - A Guide to the Theory of NP-Completeness*. Freeman, San Francisco, CA.
[10] Howe A.E., Dahlman E. (2002) A Critical Assesment of Benchmark Comparison in Planning. *Journal of Artificial Intelligence Research*, 1-33.
[11] Joslin D., Roach J. (1989/90) A Theoretical Analysis of Conjuctive-Goal Problems. *Artificial Intelligence*, 41, 97-106.
[12] Klir, G.J., T.A. Folger. 1988. *Fuzzy Sets, Uncertainty, and Information*. Prentice-Hall International, Inc. USA.
[13] McDermott D., Hendler J. (1995) Planning: what it is, what it could be, an introduction to the special issue on planning and scheduling. *Artificial Intelligence*, 76, 1-16.
[14] Nebel B., Koehler J. (1995) Plan reuse versus plan generation: a theoretical and empirical results. *Artificial Intelligence*, 76, 427-454.
[15] Nilson N.J. (1980) *Principles of Artificial Intelligence*. Toga Publishing Company, Palo Alto, California.
[16] Popovic D., Bhatkar V.P. (1994) *Methods And Tools For Applied Artificial Intelligence*. Marcel Dekker, Inc., New York, NY.
[17] Rintanen J. (1999) Constructing Conditional Plans by a Theorem-Prover. *Journal of Artificial Intelligence Research*, 10, 323-352.
[18] Ruspini E.H., Saffiotti A., Konolige K. (1995) Progress in Research on Autonomous Vehicle Motion Planning, in: Yen, J.; R. Langari; L.A. Zadeh. *Industrial Applications of Fuzzy Logic and Intelligent Systems*. IEEE Press. New York.
[19] Sierocki I. (1997) A Serial Decomposition of Planning Problems. *Proc. Fourth International Symposium on Methods and Models in Automation and Robotics*. Międzyzdroje, Poland, 1179-1184.
[20] Slaney J., Thiebaux S. (2001) Block World revisited. *Artificial Intelligence,* 125, 119-153.
[21] Slavin T. (1996) Virtual port of call, *New Scientist*, June 1996, 40-43.
[22] Smith, D.E., Weld D.S. (1998) Conformant Graphplan. *Proc. 15th National Conf. on AI,* www.aaai.org.
[23] Świerniak, A., Gałuszka A. (1999) Betweenness and Indistinguishability in Modeling and Control of Uncertain Systems, *Proc. of 18th IASTED Conference Modeling, Identification and Control*, Innsbruck, 56-58.
[24] Weld, D.S. (1999) Recent Advantages in AI Planning. *AI Magazine*.
[25] Weld, D.S., Anderson C.R., Smith D.E. (1998) Extending Graphplan to Handle Uncertainty & Sensing Actions. *Proc. 15th National Conf. on AI*, www.aaai.org, 897-904.

Instytut Badań Systemowych
Polskiej Akademii Nauk

ISBN 83-89475-01-4