4K1-6

ドメイン非依存プランニングアルゴリズムとドメイン依存アルゴリズムの性能比較に関するケーススタディ

A Case Study for Performance Comparisons of Domain Independent Planning Algorithms and Domain Dependent Algorithms

今井 達也[*1][*2]
Tatsuya Imai

[*1]東京工業大学 大学院情報理工学研究科 数理・計算科学専攻
Department of Mathematical and Computing Sciences, Graduate School of Information Science and Engineering, Tokyo Institute of Technology
[*2]日本学術振興会　特別研究員
Research Fellow of Japan Society for the Promotion of Science

ドメイン非依存プランニングアルゴリズムは，理論的には，ある種のクラスのあらゆるプランニング問題を解くことができるアルゴリズムである．しかし，一般に汎用性と効率はトレードオフの関係にあり，実用的な性能を有するドメイン非依存アルゴリズムは未だ開発されていない．本研究では，最小シュタイナー木問題について，ドメイン非依存な輸送型古典的プランニングアルゴリズムとドメイン依存アルゴリズムの性能比較を行う．

1. はじめに

プランニング問題とは，与えられた目的を達成するための活動の手順を推論する問題であり，機械学習や自然言語処理などと並んで人工知能に関する主要な問題の一つである．例えば，スライディングパズルやルービックキューブなどの盤面が一つ与えられたときに，そのパズルの解を求める問題はプランニング問題である．この他にも，荷物輸送問題やスケジューリング問題，ロボットの制御などといった数多くの実用的な問題もプランニング問題に含まれる．特に，**ドメイン非依存プランニング**の研究では，ある特定の問題を解くアルゴリズムではなく，ある種のクラスのあらゆるプランニング問題を効率よく解ける単一のアルゴリズムの開発が目標とされている [3].

本稿では，**輸送型古典的プランニング** [4] と呼ばれるドメイン非依存プランニングクラスに対する最適解探索アルゴリズムに焦点をあて，そのアルゴリズムの**最小シュタイナー木問題**に対する性能を論じていく．

輸送型古典的プランニングは，プランニング研究者が広く研究しているクラスの中で最も単純な**古典的プランニング**の部分クラスの一つである．古典的プランニングの充足解探索問題は PSPACE 完全であることが知られており [2]，コンペティションなどを通じて性能改善が盛んに行われてはいるものの，実用上十分な性能を有するソルバは未だ実現されていない．そこで著者は，ある程度の性能効率とある程度の汎用性を両立させるべく，古典的プランニングよりも更に小さなクラスでのアルゴリズム開発を提言し，輸送型古典的プランニングやその拡張である荷物容量付き輸送型古典的プランニングのためのドメイン非依存最適解探索アルゴリズムを提案した [6, 7].

ドメイン非依存プランニングアルゴリズムを実用化するためには，アルゴリズムが対象のプランニングクラスの中の個々のドメインの問題をどの程度効率的に解くことができるかが問題となる．表 1 に示すように，[6] のアルゴリズムは一般的な古典的プランニングのためのドメイン非依存アルゴリズムよりも高速に多くの輸送型古典的プランニング問題を解くことができる．しかし，このアルゴリズムが輸送型古典的プランニングの中のそれぞれのドメイン依存アルゴリズムと比べてどれくらいの解答能力を有しているかは知られていない．そこで本研究では，輸送型古典的プランニングが取り扱うことのできる問題ドメインの一つである無向グラフ上の最小シュタイナー木問題に焦点を当て，[6] の手法を更に拡張したアルゴリズム（特殊化ではないことに注意せよ）の性能を評価した．

連絡先: imai7(AT)is.titech.ac.jp

表 1: 輸送型古典的プランニングのベンチマーク問題に対する解答問題数．1440 問のベンチマーク問題に対し，一問あたり 2 GB のメモリ制限と下記の時間制限を設けて，制限以内に解き終えた問題の数を数えた．D はダイクストラ法．$h_{\max}$ は古典的プランニングに対する既存のドメイン非依存アルゴリズム（A* と $h_{\max}$ ヒューリスティック [1]）．h_{app} は [6] の提案手法．実験環境には後述の計算機実験と同じ環境を用いた．

	D	$h_{\max}$	h_{app}
1 分	429	513	786
10 分	429	520	809
30 分	429	520	809

次節では本稿で使用する概念の定義を行う．3 節では輸送型古典的プランニング問題の性質および [6] のアルゴリズムの概略を述べる．4 節では提案アルゴリズムの構成を述べる．5 節では，提案アルゴリズムの性能評価，および，最小シュタイナー木問題に対する既存のアルゴリズムと提案アルゴリズムとの簡単な性能比較について論じていく．

2. 準備

2.1 無向グラフ上の最小シュタイナー木問題

無向グラフ上の最小シュタイナー木問題は，

- コスト付き無向グラフ $G = \langle V, E, c\rangle$．$V$ は頂点集合，E は辺集合，$c : E \to \mathbb{Q}_0^+$ は辺コスト関数．
- **要求点の集合** $R \subseteq V$.

が与えられたときに，R の連結成分を含む G の部分グラフの中で，グラフ中に含まれる辺コストの和が最小になるものを求める問題である．この問題は NP 困難であることが知られている．以下ではこの問題を単に最小シュタイナー木問題と呼ぶ．

2.2 輸送型古典的プランニング問題

輸送型古典的プランニング問題は，無向グラフで表される**地図**の上で，**輸送機**を操って**荷物**を目的地に届ける問題である．

形式的には，**輸送型古典的プランニング問題**は．次の十つ組 $\mathcal{T} = \left\langle G, M, P, \mathrm{cap}, c_m, c_p, l_{M_0}, l_{P_0}, f_0, l_* \right\rangle$ によって与えられる．以下ではこれを単に輸送型問題と呼ぶ．

- 無向グラフ $G = \langle V, E \rangle$：**地図**．
- M：**輸送機**の集合．
- P：**荷物**の集合．
- $\mathrm{cap} : M \to \{1, \cdots, |P|\}$：**容量**．
- $c_m : E \to \mathbb{N}_0$：**移動コスト**．
- $c_p : P \to \mathbb{N}_0$：**積み降ろしコスト**．
- $l_{M_0} : M \to V$：**輸送機の初期位置**．
- $l_{P_0} : P \to V \cup M$：**荷物の初期位置**．
- $f_0 : M \to F$：**初期燃料**．
- $l_{P_*} : P \to V$：**目標位置**．

ただし，$F = \{0, \cdots, 4|V||P|\} \cup \{\infty\}$ である．

輸送型問題の**状態**は，三つ組 $\langle l_M, l_P, f \rangle$ によって定義される．$l_M : M \to V$，$l_P : P \to V \cup M$，$f : M \to F$ はそれぞれ**輸送機の位置**，**荷物の位置**，**燃料**と呼ばれる関数である．

輸送型問題は，**初期状態** $\left\langle l_{M_0}, l_{P_0}, f_0 \right\rangle$ を目標状態 $\langle l_M, l_{P_*}, f \rangle$（ただし l_M, f は任意の輸送機の位置および燃料）に遷移させる最小コストの行為列（＝**プラン**）を求める問題である．**行為**は状態を別の状態に遷移させる作用素であり，各々コストを持っている．プランのコストは各行為のコストの和によって定義される．状態 $\langle l_M, l_P, f \rangle$ に対して適用できる行為は高々次の三種類である．ただし行為適用後の状態を $\left\langle l'_M, l'_P, f' \right\rangle$ とおく．

- $\mathrm{move}(m, v)$：$m \in M$ を $l_M(m) \in V$ から $v \in V$ に移動させる（$l'_M(m) := v$）．$e = \{l_M(m), v\} \in E, f(m) \neq 0$ のとき適用可能．$f_0(m) \neq \infty$ のときは $f'(m) := f(m) - 1$．コスト $c_m(e)$．
- $\mathrm{push}(m, p)$：$m \in M$ に $p \in P$ を積み込む（$l'_P(p) := m$）．$l_P(m) = l_P(p)$ かつ $|\{p' \in P \mid l_P(p') = m\}| < \mathrm{cap}(m)$ のとき適用可能．コスト $c_p(p)$．
- $\mathrm{pop}(m, p)$：$m \in M$ から $p \in P$ を降ろす（$l'_P(p) := l_M(m)$）．$l_P(p) = m$ のとき適用可能．コスト $c_p(p)$．

3. 輸送型問題の特徴および既存解法

3.1 輸送型問題が表現可能なドメイン

輸送型古典的プランニングが取り扱うことのできるドメインは，一般的な古典的プランニングと比べると狭いものの，集合被覆問題や最小シュタイナー森問題，更に一部の車両配送問題などを表現できることが知られている（証明は [5] の近似困難性の証明等に依る）．例えば，最小シュタイナー木問題 $\langle G = \langle V, E, c \rangle, R \rangle$ は，次の輸送型問題 $\left\langle G, M, P, \mathrm{cap}, c_m, c_p, l_{M_0}, l_{P_0}, f_0, l_* \right\rangle$ の最適解における輸送機の軌跡を合成したグラフを求める問題に帰着することができる．証明は紙面の都合で省略する．

- 地図および移動コストには元の $\langle V, E, c \rangle$ を用いる．
- G の各頂点 $v \in V$ について，v の次数と同じだけ初期燃料 1 かつ容量無限大で初期位置が v の輸送機を配置する．
- $|P| = |R| - 1$ 個の荷物を用意し，$r_0 \in R$ を一つ固定して全ての荷物の初期位置を r_0 とする．各荷物の目標位置はそれぞれ $R \setminus \{r_0\}$ とし，積み降ろしコスト c_p は 0 とする．

3.2 輸送型問題の計算困難性

一般的な輸送型問題は NP 困難であり，また P ≠ NP ならば多項式時間近似不可能であることが知られている．ただし，この困難性は燃料が有限であることに起因し，全ての輸送機の燃料が無限であるような輸送型問題については定数近似アルゴリズムが知られている．

3.3 一般的な輸送型問題に対する既存の解法

本節では [6] のアルゴリズムを簡単に述べる．紙面の都合でアルゴリズムの全てを詳細に述べることができないため，不明瞭な点に関しては [6] を参照していただきたい．

[6] では，一般的な古典的プランニング問題の解法と同様に，輸送型問題を**最短経路探索問題**に帰着し，ヒューリスティック探索で解く手法を用いている．輸送型問題において，行為による状態の遷移は，状態を頂点とし行為を辺とする辺コスト付き有向グラフによって表すことができる（輸送型問題の地図や最小シュタイナー木問題のグラフとは異なるものであることに注意せよ）．これに対して初期状態に対応する頂点から目標状態に対応する頂点への最短経路を求めれば，その経路に対応する行為列がそのまま元の輸送型問題の最適解となる．ただし，状態遷移のグラフは一般的に問題の記述長に対して指数的以上に大きいため，ダイクストラ法のような全探索的なアルゴリズムでは到底効率的に解くことができない．そこで，少しでも効率的に探索を行うためにヒューリスティック探索法の一種である **A* アルゴリズム**を用いている．

A* を使用するためには**許容的ヒューリスティック**と呼ばれる関数が必要である．許容的ヒューリスティックは最短経路を探索するグラフ上の頂点 v を引数に取り，v からその再近傍の目標頂点への最短経路のコストの下界を返す関数である．許容的ヒューリスティックには，与えられた頂点を利用して元の問題より簡単な別の最適化問題を作り，最適解や近似解を計算してそのコストを加工した値を返す関数が使用されることが多い．

[6] では一般的な古典的プランニングと同様に，**削除効果緩和問題**の解を計算する関数を許容的ヒューリスティックに使用している．輸送型問題の状態 S の削除効果緩和問題は，与えられた輸送型問題を以下のように変化させた，S を初期状態とする古典的プランニング問題である．

- 輸送機や荷物は複数の位置に同時に存在でき，行為はそれらの位置を集合和的に重ね合わせていく．例えば $m \in M$ について $l_M(m) = \{u\}$ が成立している状態 $\langle l_M, l_P, f \rangle$ に行為 $\mathrm{move}(m, v)$ を適用すると，次状態の輸送機の位置 l'_M では $l'_M(m) = \{u, v\}$ となる．
- 全ての輸送機の容量は無限大．
- 全ての輸送機の燃料は無限大．

なお，削除効果緩和問題の最適解のコストは常に元問題の最適解のコスト以下であるという性質が知られている [1]．

輸送型問題の削除効果緩和問題は NP 困難であるが，多項式時間 4 近似が可能である [6]．[6] では，与えられた状態 S に対して，S を初期状態とする削除効果緩和問題を作り，その 4 近似解を計算し，その解のコストを 1/4 倍した値を返す関数を許容的ヒューリスティックに使用している．

4. 提案アルゴリズム

3.1 節で述べた輸送型問題を 3.3 の手法で解けば，原理的には任意の最小シュタイナー木問題を解くことができる．しかし，本研究では更に効率よく最小シュタイナー木問題を解くために，輸送型古典的プランニングを拡張したクラスを定義し，このクラスに対するアルゴリズムを提案した．

4.1 輸送型問題の状態数

まず，本節では 3.1 の手法で最小シュタイナー木問題を帰着して作った輸送型問題の状態数を観察する．最小シュタイナー木問題 $\langle G = \langle V, E, c\rangle, R\rangle$ を帰着した輸送型問題 $\left\langle G, M, P, \mathrm{cap}, c_m, c_p, l_{M_0}, l_{P_0}, f_0, l_*\right\rangle$ の状態数は，各輸送機の位置が最悪 $|V|$ 通り（G が完全グラフのとき）で荷物の位置が $|V|+|M|$ 通りであることから，最悪で

$$\begin{aligned} |V|^{|M|}(|V|+|M|)^{|P|} &= 2^{|M|\log|V|+|P|\log(|V|+|M|)} \\ &= 2^{2|E|\log|V|+(|R|-1)\log(|V|+2|E|)} \end{aligned} \quad (1)$$

である（$|M| = 2|E|$，$|P| = |R|-1$ より）．一方，例えば辺に関する分枝限定法によって最小シュタイナー木問題を解く場合，生成木の最悪の大きさは $2^{|E|}$ である．毎回最悪の探索を行われるとは限らないものの，各アルゴリズムの最悪の計算効率は探索空間の大きさに比例する．この最悪時の探索空間の大きさの違いを埋めるため，本研究では新たなアルゴリズムを提案した．

4.2 削除効果緩和問題を用いた提案アルゴリズム

実は，最小シュタイナー木問題 $\langle G = \langle V, E, c\rangle, R\rangle$ は次の輸送型問題の削除効果緩和問題 $\left\langle G, M, P, \mathrm{cap}, c_m, c_p, l_{M_0}, l_{P_0}, f_0, l_*\right\rangle$ の最適解の輸送機の軌跡を合成したグラフを求める問題に帰着できることが知られている [6].

- 地図および移動コストには元の $\langle V, E, c\rangle$ を用いる．
- $r_0 \in R$ を一つ固定して，初期位置を r_0 とする燃料・容量無限大の輸送機 m を一つだけ用意する．
- $|P| = |R|-1$ 個の荷物を用意し，全ての荷物の初期位置を m とする．各荷物の目標位置はそれぞれ $R \setminus \{r_0\}$ とし，積み降ろしコスト c_p は 0 とする．

通常の $2|E|$ 個の輸送機の代わりに位置重ね合わせの輸送機を一台使うことによって，この輸送型問題の状態数は $2^{|V|}(|V|+|M|)^{|P|}$ となる．更に，簡単な枝刈り（中途半端な位置で荷物を降ろさない等）によって状態数を $2^{|V|}+|P|$ にすることができる．

輸送型プランニングの削除効果緩和それ自体が輸送型プランニングと同様の広さのドメイン非依存プランニングクラスとなるかは明らかではない．しかし，輸送型問題に位置重ね合わせの輸送機の使用を許すことによって，元の輸送型問題の性質を維持したまま，最小シュタイナー木問題をより効率的に取り扱うことができるようになる．以下ではこの問題を**拡張輸送型問題**と呼ぶ．拡張輸送型問題の削除効果緩和問題は，単なる輸送型問題の削除効果緩和問題なので，3.3 節のアルゴリズムはそのまま拡張輸送型問題に使用することができる．荷物の積み降ろしの枝刈りに関しては次の補題が成り立つ．

補題 1 解を持つ任意の拡張輸送型問題について，初期状態で燃料・容量無限大の位置重ね合わせ輸送機に積まれている荷物に，それぞれ一度しか pop を適用せず，またこれらの pop が最後にまとめて現れるような最適プランが存在する．

証明：上記の荷物の集合を P とおき，初期状態で $p \in P$ を積んでいる輸送機を $m(p)$ とおく．まず，P のいずれかの要素に二度以上 pop を適用する，すなわち，ある $p \in P$ をその目標位置以外で $m(p)$ とは別の輸送機 m' に積み替えるようなプランを一つ固定する．このプランで p の積み替え以降に登場する全ての m' を $m(p)$ に置き換えた行為列は，$m(p)$ が燃料・容量無限大の位置重ね合わせ輸送機であることから，正しい実行可能プランである．行為のコストが輸送機に依存しないため，このプランのコストは m' を使う場合と等価である．よって，P の要素に一度ずつしか pop を適用しない最適解が存在する．

それぞれの $p \in P$ について $m(p)$ の容量が無限大であることから，p が他の荷物の移動の障害になることもない．すなわち，P の各要素はいつ降ろしてもよいため，P に対する一度の pop を最後にまとめて行うような最適解が存在する．

5. 提案アルゴリズムの性能評価

5.1 計算機実験による性能評価

本研究のために作成した最小シュタイナー木問題のベンチマーク問題集を利用して，提案アルゴリズムがどの程度の規模の問題を現実的な時間で解くことができるのかを評価した．性能の評価には問題一問あたり 10 分の時間制限と 2 GB のメモリ制限を設けて，制限以内に解が求まった問題を「解けた」と判定して，解けた問題の数を数えた．

5.1.1 実験環境

本計算機実験には東京工業大学の TSUBAME 2.0 を用いた*1．ノード・プロセス・スレッド間の一切の並列化は行わず，一つの問題の解答には一つのコアを使用した．それぞれのアルゴリズムは C++ 言語によって一から実装した．使用したコンパイラは gcc 4.3.4 で，オプションは -O3 -DNDEBUG のみである．

5.1.2 各種ドメイン非依存プランニングアルゴリズムの評価

本研究ではまず各種ドメイン非依存プランニングアルゴリズムの最小シュタイナー木問題に対する性能評価として，表 2 に示す八種類のアルゴリズムの解答能力を計測した．使用した最小シュタイナー木問題は頂点数がそれぞれ $10, 20, \cdots, 990$，要求点数がそれぞれ $2, 3, \cdots, 10$ の計 891 問である．グラフは，辺の本数の期待値が頂点数の 3 倍になるようにパラメータを定めたランダムグラフである．本実験では，グラフの作成には単一の乱数種のみを用いた．辺のコストは全て 1 とおいた．な

表 2: 各種アルゴリズムの解答問題数．D^{sup} および $h^{\mathrm{sup}}_{\mathrm{app}}$ は 4.2 節の帰着を用いてそれぞれダイクストラ法および [6] の提案手法で解く方法．c_1 は補題 1 の枝刈りを許容的ヒューリスティックに対して実装した方法，c_2 は補題 1 の枝刈りを後者状態生成の時点で行うよう実装した方法である．すなわち c_1 の状態数は c_2 の状態数のおよそ要求点の個数倍となる．$h_{\max}$ および h_{app} は 3.1 節の帰着を用いて表 1 とそれぞれ同様のアルゴリズムで解く手法である．

	D^{sup}	D^{sup}, c_1	D^{sup}, c_2	$h^{\mathrm{sup}}_{\mathrm{app}}$	$h^{\mathrm{sup}}_{\mathrm{app}}, c_1$	$h^{\mathrm{sup}}_{\mathrm{app}}, c_2$	$h_{\max}$	h_{app}
#	100	130	144	180	270	303	20	20

お，[6] で使用したベンチマーク問題集を利用して各アルゴリズムの一般の輸送型問題での解答能力も調査したが，枝刈りアルゴリズムを加えても D^{sup} は D と，$h^{\mathrm{sup}}_{\mathrm{app}}$ は h_{app} と等価な解答能力を発揮した．数表は紙面の都合で省略する．

*1 環境に関する詳細な説明は http://tsubame.gsic.titech.ac.jp/hardware-architecture 等を参照．

次に提案アルゴリズムをより詳細に評価するため，追加のベンチマーク問題を用いて $h_{\mathrm{app}}, \mathrm{c}_2$ の解答能力を精査した．ベンチマーク問題には，表 2 の実験に使用した各問題の期待辺数を 3, 4, 5 の 3 種類，辺コストの最大値を 0, 10, 100, 1000 の 4 種類（各辺のコストは 0 から 最大値の間で一様ランダムに決まる），乱数種の値を 4 種類に増やした，計 $891 \times 48 = 42768$ を使用した．図 1 にグラフの頂点数および要求点数別の解答問題数のカラーマップを示す．

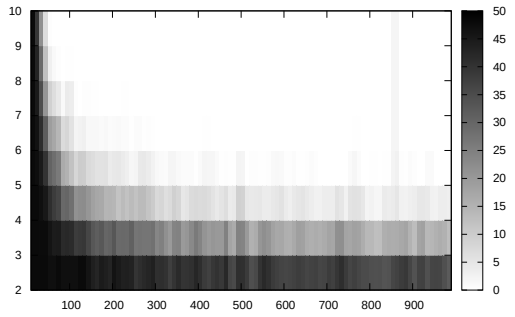

図 1: $h_{\mathrm{app}}, \mathrm{c}_2$ の解答問題数．全解答問題数は 11028.

5.2 ドメイン依存アルゴリズムとの性能比較

本研究では，著者の実験環境上で最小シュタイナー木問題のドメイン依存アルゴリズムのソルバを走らせることはしなかった．理由は以下の二点である．

まず一つ目の理由は，2013 年 3 月現在，インターネットで容易に手に入る最小シュタイナー木問題のソルバソフトウェアが事実上一つも無いことである．Google で「steiner tree solver」「steiner forest program」等のキーワードで検索をかけると，上位数十件が最小シュタイナー木問題のアルゴリズムに関する論文や書籍で占められた．ソルバを提供していたのであろうサイトが一つだけ挙げられるが，ソルバをダウンロードするためのリンクはリンク切れになっていた．後でより具体的な議論を行うが，このような問題を解決できることがドメイン非依存プランニングを研究する大きな意義の一つである．

もう一つの理由は，提案アルゴリズムの性能が（輸送型問題に対して既存のドメイン非依存プランニングアルゴリズムよりも遥かに優れているのにも関わらず），１０年以上前の論文の手法に追いついていないことである．表 3 は [8] の Table 5 からの一部抜粋である．ただし，[8] の手法は，最小シュタイナー木問題の性質を利用して，探索を始める前に問題の縮小および大規模な枝刈りを行うアルゴリズムである．表 3 の問題はいずれも前計算だけで解が求められた問題である．

表 3: [8] の提案アルゴリズムの問題解答時間．各インスタンスは OR-Library からの引用である．

名前	頂点数	要求点数	辺数	時間（秒）
C11	500	5	2500	0.06
C12	500	10	2500	0.04
C13	500	83	2500	0.10
C14	500	127	2500	0.04
C15	500	250	2500	0.03

6. まとめ

輸送型古典的プランニングのための新たなドメイン非依存プランニングアルゴリズムを提案し，最小シュタイナー木問題に対する解答能力を計算機実験によって評価した．提案アルゴリズムは既存のドメイン非依存アルゴリズムと比べて非常に効率的に最小シュタイナー木問題を解くことができたが，ドメイン依存アルゴリズムの代わりに使用するには未だ性能が不十分であった．

枝刈り c_1 と c_2 の性能差や 5.2 節の議論から，探索を始める前に可能な限り問題を縮小することがアルゴリズムへの性能向上につながるのではないかと考えられる．

なお，最小シュタイナー木問題のようなソルバソフトウェアの不在は，他の数多くの組み合わせ最適化問題でも同様に起こるものであると推測される問題点である．アルゴリズムに関する研究論文を理解すればソルバを自前で実装することもできるが，論文の理解およびソフトウェアの実装は，とりわけエンドユーザーにとっては，どちらも多大な労力を要するという難点がある．場合によっては対象の問題に対するドメイン依存最適アルゴリズムが無い，あるいは解きたい問題が一般的に認知されている問題ではないことすらありえるだろう．一方，ドメイン非依存プランニングにおいては，様々なクラスのための多数のソルバソフトウェアを容易に入手することができる．ドメイン非依存プランニングアルゴリズムを活用するために唯一不足しているのが計算性能でなのである．ドメイン依存アルゴリズムとの性能差を埋め，ドメイン非依存プランニングアルゴリズムの普及に努めていくことが今後の課題として考えられる．

謝辞

本研究は JSPS 特別研究員奨励費（課題番号 24・8506）の助成を受けたものです．

参考文献

[1] Bonet, B. and Geffner, H.: Planning as heuristic search, *Artificial Intelligence*, Vol. 129, No. 1, pp. 5–33 (2001)

[2] Bylander, T.: Complexity Results for Planning, *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence*, Vol. 1, pp. 274–279 (1991)

[3] Ghallab, M. and Nau, D. and Traverso, P.: *Automated Planning: Theory & Practice*, Morgan Kaufmann, (2004)

[4] Helmert, M.: On the Complexity of Planning in Transportation Domains, *Proceedings of the Sixth European Conference on Planning*, pp. 349–360 (2001)

[5] Helmert, M.: *Understanding Planning Tasks*, Springer, pp. 239–251 (2008)

[6] 今井: 輸送型古典的プランニングにおける緩和問題の解析と近似アルゴリズム, 第 88 回人工知能基本問題研究会, (2013)

[7] 今井: 荷物容量付き輸送型古典的プランニング問題に対する許容的ヒューリスティック, 第 89 回人工知能基本問題研究会, (2013)

[8] Polzin, T., and Daneshmand, S. V.: Improved algorithms for the Steiner problem in networks, Discrete Applied Mathematics, (2001)