1 はじめに

WWW の急速な広がりに伴い，インターネットを通してアクセスできる情報は，急激に増加している．このような状況で，WWW にある有用な Web ページを見ることにより，“ある概念（**目標概念**と呼ぶ）を理解する” という作業が，多くのユーザにより日常的に行われている．しかし，この作業において，ユーザは，目標概念を理解するために有用な Web ページが，WWW 上のどこにあるのかを知らない．よって，ユーザは，その有用なページを何らかの方法により，探索する必要が生じる．さらに，ひとつの Web ページが，目標概念を理解するために必要十分な情報を含んでいない場合も多く，その場合複数の有用な Web ページを探索する必要がある．また，検索して得られた Web ページ中に，さらにユーザが知らない概念が含まれている場合も多々あり，さらに再帰的に有用な Web ページの探索を繰り返していかなければならない．このような作業は，現状の WWW の技術では，多くの時間を必要とするし，また試行錯誤を含むため，ユーザにとって大きな負荷となる．また，この概念理解に有用な Web ページを探す作業は，goo，MetaCrawler などの検索エンジンや，Web ロボット [3] を利用することにより，部分的には効率化，自動化できるが，結局検索されたページを見てチェックするところはユーザがやらねばならず，本質的な解決にはならない．

本研究において，我々は，概念理解のための有用な Web ページの探索を人工知能におけるプラニングとして捉え，プラニングの枠組を適用することにより，この作業全体を自動化することを目的とする．ここで，プランとは，目標概念を理解するために，ユーザが見るべき Web ページの系列である．このように，目標概念に対し，それを理解するのに必要十分な Web ページの系列を生成するプラニングを，**ナビゲーションプラニング**（navigation planning）と呼ぶ．ナビゲーションプラニングでは，一つの Web ページを見て理解することを，プラニングにおける一つの行為として捉え，その行為を **U-オペレータ**として定義する．ただし，このような定義では，従来のプラニングのように事前に必要な U-オペレータを全て用意しておくことは，世界中のすべての Web ページに対し予め U-オペレータを作っておくことに対応するため，現実的に不可能である．よって，ナビゲーションプラニングは，プラニング過程に必要に応じて，Web ページから U-オペレータを自動生成する．WWW における概念理解をプラニングとして扱った点，プラニング中にオンラインで U-オペレータ生成を行う点において，ナビゲーションプラニングは，新規性を有する．

Softbot[5] は，欲しい情報を集めるために必要なコマンドの系列を自動生成する完全な半順序プラニングシステムである．また，Occam[8] も，同様のプランニングをより効率よく行い，異種の情報源を扱えるプラナーである．Sage[7] は，分散した情報源において，プラニング，実行，再プラニング，センシングを統合するために開発されたシステムである．これらの研究の目的は，情報収集の手続きをプランとして自動生成することであり，そのプランは，UNIX コマンドやデータベースの検索コマンドの系列として表現される．対照的に，ナビゲーションプラニングの目的は，ユーザの概念理解を誘導するようなプランの生成であり，そのプランは，目標概念を理解するために，ユーザが見るべき Web ページの系列で表現される．

Web ロボット [3] も，検索エンジンの URL データベース構築のために URL 情報を自動収集する．しかし，Web ロボットは，リンクの張られている Web ページ間を探索するに過ぎず，また目的に合った制御もされない．ナビゲーションプラニングでは，後述するようにリンクが明示的に張られていない Web ページも探索することが可能であり，目標概念の理解のために適切に制御される．

情報収集における学習システムとして，ShopBot [4] がある．ShopBot は，オンラインショッピングの Web ページにおける商品の価格を記述したテキストのパターンを学習し，最も安い店を人間よりも効率的に見つけることができる．また，WebWatcher [1] と Letizia [9] は，ユーザが次に見たいであろう Web ページを予測して，提示する．ユーザのブラウジングの履歴から，予測の学習を行う．ただし，次のページを提示するだけであり，ナビゲーションプラニングのように系統立てた Web ページ系列を生成／提示するわけではない．

2 ナビゲーションプラニング

本研究では，**ナビゲーション**とは，概念理解を誘導するために，ユーザに有用な Web ページを提示することを意味する．また，その有用な Web ページの系列を**プラン**とし，そのプランを自動生成することを**ナビゲーションプラニング**と呼ぶ．

まず，ユーザが目標概念を理解するために有用な Web ページをブラウズする手続きは，以下のようにまとめることができる．この手続きは，ユーザが停止するまで繰り返される．

1. 検索エンジンを使って，目標概念に関連のある Web ページを検索する．
2. 検索された Web ページのうち，役に立ちそうなページを見て理解する．
3. その Web ページにおいて，未知の概念を目標概念として，1にいく．

この手続きは，以下のような対応により，プラニング [6][11] として定式化できる．

- **行為（action）：** Web ページに記述されている概念を理解する．
- **状態（state）：** ユーザの知識状態．既知の概念を表す単語の集合により記述．
- **初期状態（initial state）：** ユーザの初期の知識状態．
- **目標状態（goal state）：** ユーザが理解したい目標概念．目標概念を表す単語の集合により記述．
- **オペレータ（operator）：** Web ページを見て，知識を獲得するという行為を表す **U-オペレータ** U-Op(URL) は，以下の要素からなる．
  - **ラベル：** Web ページの URL でラベル付け．
  - **条件（condition）：** その Web ページを理解するために必要な知識である**条件知識** $C = \{c_1, \cdots, c_i\}$．c は，その要素の知識で，**条件語**と呼ばれる．
  - **効果（effect）：** その Web ページを理解することにより得られる知識である**効果知識** $E = \{e_1, \cdots, e_j\}$．e は，その要素で，**効果語**と呼ばれる．

条件知識と効果知識は，個々の知識を表す単語の集合で記述される．このような定式化により，プラニングの枠組 [6][11] を，ナビゲーションプラニングに適用できる．

ただし，ナビゲーションプラニングには，従来のプランニングでは扱われていない重要な問題がある．それは，あらかじめ必要な U-オペレータを用意することが不可能なことである．U-オペレータは，Web ページ毎に必要なため，それをすべて事前に用意することは，世界中の膨大な数の Web ページすべてについて，U-オペレータを生成しておくことを意味する．これは，現実的には不可能である．また，必要なページは，用意された U-オペレータのごく一部なので，多くの U-オペレータを用意しておくことは効率が悪い．よって，ナビゲーションプラニングでは，必要になった時に逐次的に U-オペレータを自動生成する方法を採る．

3 Web ページからの U-オペレータ生成

Web ページから，条件知識と効果知識を自動抽出することにより，U-オペレータを自動生成する方法について述べる．本研究では，条件語と効果語は，Web ページに記述されていると仮定し，Web ページからいかにそれらを抽出するかを考える．

3.1 タグ構造による抽出

情報検索の分野において，様々なキーワード抽出法が提案されている [12]．そのほとんどは，単語の出現頻度によるものであるが，構造化されたテキストの場合，その構造を利用するのが最も効果的なキーワード抽出法の一つである．Web ページは，通常 HTML[2] という構造化された言語で記述されているので，本研究では，条件／効果知識抽出のために，そのタグ構造（<TITLE>, <Hn>, <A HREF=...>）を利用する．

条件語の抽出　最初に考えられる条件語の候補は，他の Web ページにリンクされている単語，つまり <A HREF=URL>と</A>の間の単語である．Web ページの作者は，リンクされている単語は，その Web ページを理解するために重要であると明示的に示している．さらに，その単語を理解するために有用な Web ページが，明示的にリンクされている．

しかし，残念ながらリンクの張られた単語が，すべて重要なわけではない．例えば，商用の Web ページなら，そのスポンサー企業の Web ページにたくさんリンクが張られている場合が多い．また，すべての条件語にリンクが張られているわけでもない．よって，<A HREF=URL>タグにより得られた条件語の絞り込みとリンクの張られていない条件語の候補抽出を後述する *KeyGraph* により行う．

効果語の抽出　Web ページのタイトルは，そのページを読むことにより，ユーザが獲得する知識を表していると考えられる．よって，<TITLE>と</TITLE>の間の単語を，効果語の候補とするのは妥当であろう．同様に，見出しも，その節を読むことにより学習される知識を記述していると考えられるため，<Hn>と</Hn>の間の単語も，効果語の候補とする．

3.2 *KeyGraph* によるキーワード抽出

U-オペレータの基本となる条件／効果知識を，HTML ファイルのタグだけから生成するのは困難である．<A HREF>タグによるリンクのすべてが，その文章の条件を表わす Web ページを辿るとは限らないし，逆にすべての条件語に適切なリンクが張られているとも限らないからである．よって，ナビゲーションプラニングでは，キーワード抽出法 *KeyGraph* を併用することによって条件／効果知識を得ることにする．

KeyGraph [10] は，Web ページの主張点を表わすキーワードを高速に抜き出すことができる．*KeyGraph* では，まず単語の共起度から生成したグラフのクラスタ（単語を表わすノードと，それらの間の共起リンクで結ばれた連結グラフ）によって文章の土台となる概念を表わす．したがって，このクラス

入力

- 入力ファイル（HTML ファイル，またはプレーンテキスト）とその URL

制御パラメータ

- <TITLE>の重み: α，<Hn>の重み: β，<A HREF=URL>の重み: γ
- 条件語数: N_C
- 効果語数: N_E

出力

- U-オペレータ U-Op(URL) = (URL, 条件知識, 効果知識)

手続き

1. *KeyGraph* により，入力ファイルから条件語の候補の集合 C_C と効果知識の候補 C_E を抽出．それらの候補は，[0, 1] の重みを持つ．
2. <TITLE>と</TITLE>の間の単語を重みをαとして，C_E に追加．
3. <Hn>と</Hn>の間の単語を重みをβとして，C_E に追加．
4. C_C 中の単語で，<A HREF=URL>と</A>の間に含まれる単語の重みをγに変更．
5. C_C から重みが上位 N_C 個の単語を条件知識として，C_E から重みが上位 N_E 個の単語を効果知識として取り出して，U-オペレータを生成．また，<A HREF=URL>により抽出された条件語には，リンク先の URL を割り当てる．

図 1　**Ext-OP**: U-オペレータ生成手続き

タ内の単語を，本研究では条件語の候補とする．次に *KeyGraph* では，これらの土台を統合する役割をする単語を取り出し，これをその文章の主張と見なす．そこでわれわれは，これらの主張にあたる単語をその Web ページの効果語の元とする．これらのキーワードは，条件らしさ，効果らしさを表わす実数値と共に得ることができる [10]．

KeyGraph のもう一つの長所は，コーパスを用いずに済む点である．これが長所であるというのは，TFIDF[13] のようにコーパスを用いるキーワード抽出法では通常，コーパスをある分野に限定してシステムに与えなければならない上，WWW のように多くの Web ページが生まれては消滅する環境ではコーパス情報を絶えず更新しなければならないからである．WWW 上の一般的なサイトの分野を明確に定義するのも，コーパスの膨大な情報を日々正確に更新するのも容易な作業ではない．

ナビゲーションプラニングでは，条件知識，効果知識を，この *KeyGraph* とタグ情報の組み合わせによって抽出する．U-オペレータを生成するアルゴリズム **Ext-OP** の全体は，図 1 のようになる．また，*KeyGraph* の併用により，タグ構造のないプレーンテキストの Web ページからも，条件／効果知識の抽出が可能である．

4　プラニング手続き

ナビゲーションプラニングのプラニング手続きについて述べる．図 2，図 3 が，ナビゲーションプラニングの処理の流れである．探索としては，目標状態から後向きのビーム探索 [11] を行う（図 2）．状態ノードの展開（図 3）では，検索エンジンを使った関連 Web ページの検索と，先の手続き **Ext-OP** を用いた U-オペレータ生成が行われる．

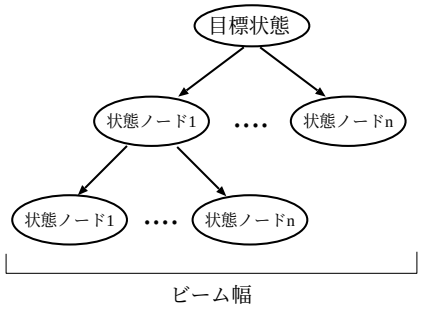

図 2　後向きビーム探索

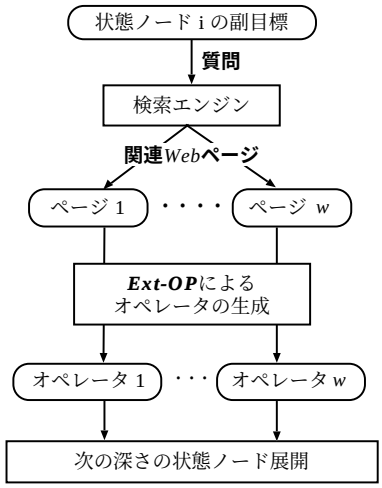

図 3　状態ノード i の展開

ナビゲーションプラニングの入出力と手続きを，図 4 と図 5 に示す．付加的な入力として，**文脈語**を用いている．文脈語は，目標概念が属する領域を表し，同じ単語でも使われる領域により意味が異なる場合に付加することで，探索の精度を上げる効果がある．

ナビゲーションプラニングの停止条件（図 5 の手続き (2)）は，後向きプラニングで一般に用いられ

入力

- 目標状態（= 目標概念）G_0：ユーザが理解したい概念を表す単語の集合.
- 文脈語 GC：目標状態の背景領域を記述する単語の集合.

制御パラメータ

- 初期状態 IS：ユーザの初期の知識状態.
- 深さ制限 l：プランの最大長.
- Web ページの制限数 w：検索エンジンにより得られる関連 Web ページの最大数.
- ビーム探索の幅 b
- 状態ノードの評価関数 H

出力

- プラン P: U-オペレータ U-Op(URL) の系列.

図 4　ナビゲーションプラニングの入出力

ているように，副目標が空になることが考えられる．しかし，この停止条件の判定には，ユーザの初期の知識状態を完全に記述しておく必要があり，これは現実的には困難である．よって，さらに停止条件として，深さの制限を入力し，その深さでプラニングを停止させることにする．

また，状態ノードの評価においては，差異をどの程度満たしているかという従来の評価に加え，副目標と U-オペレータの効果知識のもつ重みを利用できる．よって，ナビゲーションプラニングでの関数 Hは，下式のように定義する．

$$H = \sum \text{（満足された副目標の重み）} + \sum \text{（副目標を満足した効果語の重み）}$$

我々は，以上の手続きを実装し，ナビゲーションプラニングシステム *NaviPlan*を構築した．*NaviPlan*は，Perl で記述されており，ユーザとのインタフェースとして，Web ブラウザを用いる．図 6 に，目標概念が “concept formation”（概念形成）で，文脈語が “AI” の場合の *NaviPlan* の出力プラン（深さ制限 $l = 4$）を示す．Web ページ 1 は，目標概念に直接結びつくものであり，概念形成について ILP（Inductive Logic Programming）を基にして説明されている．しかし，ILP 自体についての説明はページ 1 にはない．ILP については，Web ページ 2〜4 のうち，3 で詳しく述べられ，4 には関連する機械学習の論文アブストラクトが紹介されている．実際，ユーザが，1，3，4 の Web ページを読むことによって概念形成はもとより，それと ILP，機械学習との基礎的な関係まで理解することができる．

4.1　枝刈りとキャッシング

*NaviPlan*は，図 5 の 3cにおけるファイルの取り

手続き

1. $N_0 = [n_0] = [(G_0, [])]$, $i = 0$ で初期化.（N_i は，深さ i での状態ノード系列．状態ノードは，$n_i = (G_i, P_i)$ で記述される．ここで，G_iは副目標を表す単語の集合，プラン P_iは，U-オペレータ系列.）
2. もし $i = l$，または，N_iが空なら，プラン Pを出力して終了.
3. 以下の手続きを，N_iのすべてのノード n に対して適用して，$N_{i+1} = []$ で初期化.
   (a) 差異 $D = G \cap \overline{IS}$ を抽出.
   (b) 検索エンジンに $D \cup GC$を入力して，w 個の関連 Web ページの URL を得る.
   (c) 得られた URL の http サーバとの TCP/IP 接続により，その Web ページ（HTML ファイル，またはテキストファイル）を獲得.
   (d) 得られた Web ページから，手続き **Ext-OP** を使って，U-オペレータを生成．そして，効果知識 Eと n の副目標との交わりが空でない U-オペレータに対して，下の手続きを適用.
      i. その U-オペレータの Eを nの G から取り除いたものに，U-オペレータの Cを追加することにより，G'を得る.
      ii. nのプラン Pの先頭に，その U-オペレータを加えることにより，P'を得る.
      iii. 新しい状態ノード (G', P') を N_{n+1} に追加.
4. N_{n+1}の各状態ノードにつき，評価関数 Hによる評価を行い，その上位 b 個のノードで，N_{n+1} を更新.
5. $i = i + 1$ として，(2) へ.

図 5　ナビゲーションプラニング手続き

込みに時間がかかるため，深さ 4 程度のプランの生成にも数時間を要する．よって，明かに無駄な状態ノードの枝刈りと，一度得られた html ファイルを次に使うために保存しておくキャッシングの 2 つの方法により効率化を行っている．

枝刈りについては，URL のポインタや文献のリストだけからなるページ，極端に短いかあるいは長いページなどが，図 5 の 3(d)iii で削除される．

また，キャッシングでは，図 5 の 3cで一度得られた Web ページを残しておき，次に同じページが必要な時に再利用する．これにより，*NaviPlan*を 2 倍以上に効率化することが可能になる．

5　*NaviPlan*の実験による評価

5.1　U-オペレータ生成の精度

*NaviPlan*の精度を支配するのは，U-オペレータである．ここでは予備実験として，得られる U-オペレータがどの程度条件知識と効果知識を正しく表

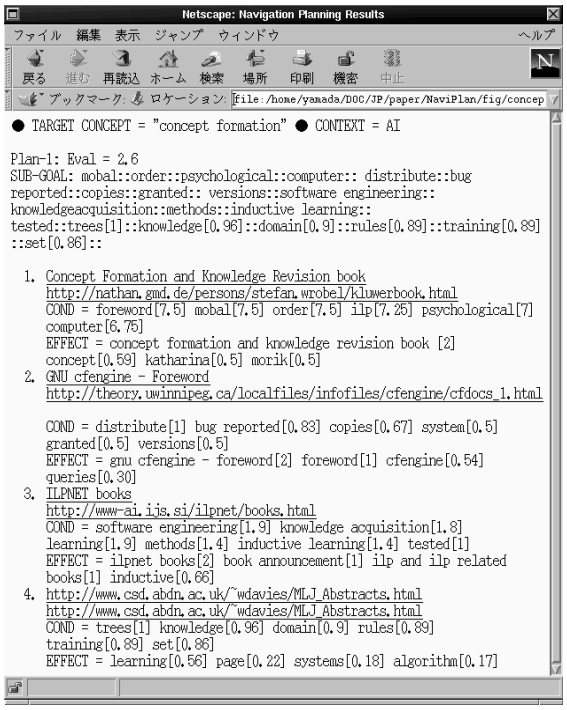

図 6 “Concept formation” のプラン

現しているかを評価する実験を行った．実験方法は，以下のとおりである．

まず，ある目標概念について得られたプラン中の 1 つの Web ページ Doc をとる．Cと Eをそれぞれ，Doc から $NaviPlan$で得られた条件知識と効果知識とする．また，c と e をそれぞれ，真の条件知識と真の効果知識とする．この予備実験では，C, E中の各単語が，c, e に含まれるかどうかを調べた．そして，C, E, $c \cap C$, $e \cap E$, $c \cap E$, $e \cap C$ の各集合の要素数を求めた．

23 の Web ページについて，Cまたは Eに選ばれた個々の単語について評価したところ，結果は $|C| = 494$, $|E| = 331$, $|c \cap C| = 367$, $|e \cap E| = 228$, $|c \cap E| = 25$, $|e \cap C| = 15$ となった．これらの値から $|c \cap C|/|C| = 0.74$, $|e \cap E|/|E| = 0.68$ を得るが，この二つの値がそれぞれ Cと Eの精度を意味する．

一方，もしキーワードを全くランダムに $|C|$ 個と $|E|$ 個のキーワードからなる二つの集合 Cと Eに分けていたら，その中には $|c \cap C|/|C| = |c|/|C \cup E| = 0.47$, $|e \cap E|/|E| = |e|/|C \cup E| = 0.29$ という割合しか真の条件キーワードと効果キーワードが含まれなかったことになる．したがって上の結果は，われわれの U-オペレータにおいて条件知識と効果知識という狙いに沿って分類できたことを示している．

5.2 *NaviPlan* の性能評価

次に *NaviPlan* の性能を，実際にシステムを用いた結果から評価した．*NaviPlan*でのノードの展開（図 3 ）における検索エンジンと以下の実験で比較に用いる検索エンジンとして，現在最も広範囲に検索可能な検索エンジンの一つである MetaCrawler（http://www.metacrawler.com/）を用いた．

5.2.1 実験方法

われわれは，以下の 5 つの手法を比較した．

手法 1: 検索エンジンのみ 目標概念と文脈語を検索キーとして入力する．出力ページとしては，ゴールを含んでいて文脈語に関係の深い Web ページを得る．そして，ユーザは出力された Web ページを，ソートされた優先順に目標概念を理解するまで読み進める．

手法 2: 検索エンジン＋リンクの利用 手法 1 と同様の入力と出力ページをまず得る．次にユーザは，手法 1 と同様に優先順位に従って Web ページをゴールを理解するまで読み進めるが，途中でページ中のリンクに従って他のページに移っても良い．

手法 3: NaviPlan ユーザは，*NaviPlan*に目標概念と文脈語を入力する．そして，*NaviPlan*の出力順（読み進めるべきと *NaviPlan*が判断した順番）に目標概念を理解するまで読み進める．

手法 4: NaviPlan＋リンクの利用 ユーザは，手法 3 と同様の入力を *NaviPlan*に与え，その結果として得られたページを，手法 3 に加えて手法 2 と同様のリンク利用を行いながら目標概念を理解するまで読み進める．

手法 5:リンクの利用 ユーザは，MetaCrawler で 1 位にランクされた Web ページだけから，目標概念を理解するまでリンクをたどっていく

この実験において設定した *NaviPlan*のパラメータは，図 5 では，$IS = []$, $l = 4$, $w = 8$, $b = 4$ で，図 1 では，$\alpha = 2$, $\beta = 1$, $\gamma = 1$, $N_C = 6$, $N_E = 4$ とした．一度のプランニングに要した時間は 2 ～ 3 時間であったが，そのほとんどの時間は WWW サーバとの通信による HTML ファイルの獲得に費やされた．よりステップ数の多いプランについても実験を行ったが，プランがほとんど得られなかったため，以下では評価していない．

5.2.2 実験結果と評価

43 通りの検索キーについて上記の 5 つの手法を比較した結果，手法 1 と手法 3 については，ともに 26 通りの目標概念が正しく理解された．リンク

利用を単独で用いた手法 5 の性能は最も低く 13 通りの目標概念が理解されただけであったが，リンク利用を手法 1 と手法 3 と併用した手法 2 と手法 4 ではそれぞれ，33 通りと 32 通りに増加したので，リンク利用に意味があることがわかる．これらの数値だけでは MetaCrawler の出力の上位から順に読み継ぐ方法が *NaviPlan* とほぼ同等の性能を持つように見える．ユーザの感想では，MetaCrawler ではプラニングを行わない代わりに目標概念（G_o）に直接関係する Web ページを多く出力するため，その中には自分に理解できるもの（解説のような文書）も含まれていることがあり，そのせいで *NaviPlan* に劣らない性能が得られたとのことであった．

実際，MetaCrawler と *NaviPlan* ではユーザが「理解できた」という目標概念が異なっていた．即ち，MetaCrawler ではユーザの知識が目標概念を理解するまでに複数の Web ページを読み継ぐ必要がない程豊かであれば短時間で理解が可能である．しかし，ユーザの予備知識が乏しい場合は出力された Web ページをトップから順に多数読むという負担の大きな作業がユーザに課されることになる．このような場合は，基礎からより高度な内容へとユーザが Web ページを読み継げるように *NaviPlan* でプラニングを行うことのメリットが高くなるのである．

このことを実験的に立証するため，ユーザが目標概念理解までにどれだけ労力を使ったか，すなわちどれだけ多くの Web ページを読んだかを評価した．するとまず，トップにランクされた Web ページだけでユーザが理解した目標概念は，手法 1 では 6 通りであったが手法 3 では 11 通りであった．この結果は，*NaviPlan* ではオペレータが期待に沿うように得られたために出力の第 1 ページがユーザの示した目標概念とよく一致したためと考えることができる．

それならば，リンクの利用がユーザの理解を促進することは先述のとおりであるから，手法 3 とリンク利用を合わせた手法 4 によってユーザは最も少ない労力で知りたいことを理解できるのではないか．このことを手法 2 と手法 4 との比較から調べてみた．7は，リンク利用と併せた MetaCrawler と *NaviPlan*，すなわち手法 2 と手法 4 を比較したグラフである．横軸は，検索エンジン（MetaCrawler）で得られた出力ページの数（検索語がどれだけ Web で普及しているかを表わしていると考えられる）であり，縦軸は，実際に目標概念を理解できるまでに読む必要があった Web ページ数である．7において，以下の 2 つの特徴を挙げることができる．

- 手法 4 (*NaviPlan* + リンク利用) によってユーザは，目標概念を理解するまでに手法 2 (検索エンジン + リンク利用）よりも少ないページを読むだけでよい．
- 検索エンジンで得られるページ数が多いほど，手法 2 は多くのページをユーザに読ませてしまうということである．このことは，一般によく用いられる語は既にその意味が広く知れ渡っていて， 改めてその定義をする Web ページが少ないことが原因であると考えられる．すなわち，目標概念を明示的に定義する Web ページが少ない場合こそ，*NaviPlan* が行う読むべきページの誘導が意味を増すからである．

なお，手法 1 と手法 3 は，ユーザが理解できた内容についてはそれぞれ手法 2 と手法 4 と同数のページを読む必要があった．かつ理解できた場合が手法 2 と手法 4 より少ないのであるから，明らかに性能が劣るものとして 7での比較は行わなかった．

以上を総括すると，MetaCrawler による検索よりも *NaviPlan* によるプラニングがユーザの効率的な理解を助ける効果があり，リンクの利用もまたユーザの理解を促進することがわかる．

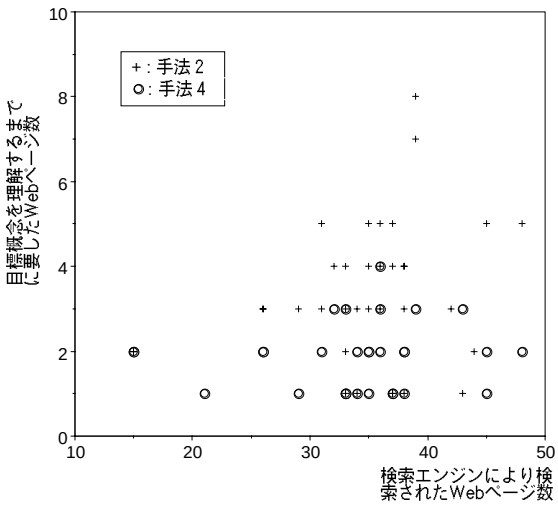

図 7　手法 2 と手法 4 の比較

6　検討課題

ナビゲーションプラニングにおける問題点について述べる．

U-オペレータの精度　プラニングの視点から言えば，*NaviPlan* は，プラニングの過程でオペレータ自動生成するところがユニークであるが，5.1でも述べたように，そのオペレータの精度が現状では十分に高いとは言えない．

この U-オペレータの精度を下げる要因は，先ず U-オペレータで用いたヒューリスティックにある．例えば，*NaviPlan* では，見出しのタグ<Hn> を利用し

て効果知識の抽出を行っているが，見出し情報を用いて得られた効果知識が信頼できないことは，[10]でも指摘されている．このヒューリスティックの精度を高めることは，*NaviPlan*の第一の検討課題である．

枝刈りの難しさ　すべてのWebページがU-オペレータ生成の対象である保障はないため，*NaviPlan*では，4.1のような枝刈りを行ってはいるが，まだまだ不十分である．現状では，無意味なWebページが含んでいる構造のパターン（リストがページの大半を占めるなど）をアドホックに用意しているに過ぎない．必要十分な枝刈り方法の確立が課題である．

プラニングの完全性と健全性　*NaviPlan*は，以下の理由により，正当なプランを必ず見つけ出すという意味での**完全性**と，見つけたプランが正当であるという**健全性**の両方を保障していない．ここで，プランが**正当である**とは，そのプランどおりにユーザがWebページをたどることにより，ユーザが目標概念を理解できることを意味する．

*NaviPlan*はビーム探索を行っているため，評価の低い状態ノードは，枝刈りにより削除される．しかし，その評価はヒューリスティックスなので，実際には正当なプランにとって必要な状態ノードを削除する場合がある．よって，完全性が保障されないことになる．

U-オペレータの精度が十分でないため，*NaviPlan*の生成したプランがユーザの知識の獲得過程を正しく記述できていない場合がある．その場合は，生成されたプランの正当性がなく，よってプラニングの健全性が保障されないことになる．

概念理解という行為の因果律　ユーザの理解は，Webページの著者にとっての条件から効果へと進む必要が本当にあるだろうかというより根本的な疑問がある．通常のプラニングでよく使われるロボットの行動のような物理的操作では，その因果律が明確であるが，U-オペレータのように“概念理解”という抽象的行為では必ずしもそうでない．例えば，行動に関する推論という方法論を，これまでにない巧緻なロボット制御に応用するというWebページがあったとする．これを効果知識（巧緻なロボット制御）について日常考え続けている研究者が読んだ場合，逆に条件知識（行動に関する推論）について学ぶことは十分に考えられる．現在われわれは，条件知識と効果知識を区別しないU-オペレータを生成した場合のプラニングについても検討している．しかし，いたずらに探索空間を広げることになりかねない．この課題は，遷移前と遷移後の状態を分けないオペレータに基づくプラニングにあたり，プラニングの分野にとっても新たな問題に挑む長期研究と考えている．

7　まとめ

ユーザが理解したい目標概念を入力するだけで，その概念理解に有用なWebページを組織的に提示することができるナビゲーションプラニングを提案し，その実装システム*NaviPlan*を構築した．このシステムは，WWW上でのユーザのブラウジングをプラニングの枠組で定式化し，必要に応じてWebページからU-オペレータを自動生成して，プラニングを行い，有用なWebページの系列をプランとして出力する．さらに，検索エンジン等との実験的比較を行い，その*NaviPlan*の有用性を示した．

参考文献

[1] R. Armstrong, D. Freitag, T. Joachims, and T. Mitchell. WabWatcher: A learning apprentice for the World Wide Web. In *The 1995 AAAI Spring Symposium on Information Gathering from Heterogeneous, Distributed Environment*. AAAI, 1995.

[2] T. Berners-Lee and D. Connolly. *Hypertext Markup Language – 2.0*. RFC 1866, 1995.

[3] F. Cheong. *Internet Agents: Spiders, Wanderers, Brokers, and Bots*. New Riders, 1996.

[4] R. B. Doorenbos, O. Etzioni, and D. S. Weld. A scalable comparison-shopping agent for the World-Wide Web. In *Proceedings of the First International Conference on Autonomous Agent*, pages 39–48, 1997.

[5] O. Etzioni and D. Weld. A SoftBot-based interface to the Internet. *Communication of the ACM*, 37(7):72–76, 1994.

[6] R. E. Fikes and N. J. Nilsson. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2:189–208, 1971.

[7] C. A. Knoblock. Planning, executing, sensing, and replanning for information gathering. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, pages 1686–1693, 1995.

[8] C. T. Kwok and D. S. Weld. Planning to gather information. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, pages 32–39, 1996.

[9] H. Lieberman. Letizia: A agent that assists Web browsing. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, pages 924–929, 1995.

[10] Y. Ohsawa, N. E. Benson, and M. Yachida. *KeyGraph*: Automatic indexing by co-occurrence graph based on building construction metaphor. In *Advanced Digital Library Conference*, 1998. to appear.

[11] S. Russell and P. Norvig. *Artificial Intelligence –A Modern Approach–*. Prentice-Hall, 1995.

[12] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. In K. S. Jones and P. Willet, editors, *Readings in Information Retrieval edt.), Morgan Kaufmann*, pages 323–328. Morgan Kaufmann, 1997.

[13] G. Salton and M. J. McGill. *Introduction to modern information retrieval*. McGraw-Hill, 1983.