Drag-and-Guess: 予測付きドラッグアンドドロップ

五十嵐 健夫　（共同研究者　西田健史）

1　はじめに

Graphical user interface (GUI) によって，コンピュータで人がこなす作業の多くは直感的かつ容易になった．しかし，単調な繰り返し作業は依然として存在し，ひとつひとつの作業がグラフィカルになったに過ぎないものが多い．そのような作業のひとつとして，電子メールを階層的なフォルダに振り分ける作業が挙げられる．毎日届く電子メールの一通一通に対してユーザは，(1) 振り分け先を決める (2) 振り分け先のフォルダを可視にする (3) ドラッグアンドドロップ（DnD）で振り分ける という単純な作業をすることになる．

このような振り分け作業の一部は自動化されているが，問題点も多い．例えば，振り分け規則を設定するのは面倒である．また，フィルタによる自動振り分けには，誤った振り分けが行われる可能性があるため全面的に信頼することができないという問題がある．ほかにも，届いたメールが読む前に振り分けられてしまうと新着メールが分散するため，それらを見逃してしまうことがある[5].

また，これらの方法は電子メールに特化しているため，ブックマーク整理のような他の振り分け作業に適用することは難しい．自動振り分けの良さを活かしながら, 簡単に安心して使える方法が望まれる．

本稿では，従来の DnD に予測提示を付加した drag-and-guess を提案する．そして，提案手法をデザインするうえでの考慮点やその評価実験について議論する．

2　関連研究

DnD を拡張した代表的な手法としてアイコン投げ[7] や drag-and-pop[3] があり，どちらの手法もドラッグ開始方向を利用している．アイコン投げはドラッグ開始方向にあるアイコンを目標アイコンだとすることで，ドロップ操作を省略する．Drag-and-pop はドラッグ開始方向にあるアイコンをカーソル付近にポップアップさせることで，大型ディスプレイにおける DnD を容易にする．これらの手法には従来の DnD と同様，ドラッグ目標が可視でなければならないという問題がある．また，小さな目標を狙うことの難しさ[2] は解決していない．それに対し提案手法は予測提示を用いることで，可視ではない目標や小さな目標への DnD を容易にする．

予測を用いたシステムは数多く存在する[6]．MailCat[4] は電子メールの振り分けを自動的に行う代わりに，各メールについて予測候補上位３つのフォルダへの振り分けを行うボタンを提示する．この手法では widget を追加する必要があるほか，アイコンの位置情報やフォルダの階層構造が失われてしまうという問題がある．それに対し提案手法では，widget を追加する必要がないうえ，アイコンの周辺情報を活用することができる．

振り分け作業の性質について調査した研究には，電子メールを題材としたもの[5] やブックマークを題材としたもの[1] がある．これらの議論は，提案手法のような「簡単な手動振り分け」の存在を前提としていない．

3　Drag-and-Guess

3.1　プロトタイプ

提案手法のデザインや評価を目的としたプロトタイプの実装を行った（図 1）．このプロトタイプは Java で実装されている．また，複数のコンポーネントにわたるイベント処理と描画のために Swing の Glass Pane 機構を利用している．

このプロトタイプはメールの振り分け作業をモデルとしていて，左から順に階層的なフォルダ，メールのリスト，メールの内容を模した３つのコンポーネントから成る．

一番右のコンポーネント上部に見られるボタンは提案手法との比較実験を目的として MailCat[4] のようなボタンを実装したものである．

図 1. プロトタイプのスクリーンショット

3.2 Drag-and-Guess

図 2 は drag-and-guess の状態遷移図である．また，図 3 は drag-and-guess 使用の様子を示している．図 3 (1) はシステムの振り分け先予測を受け入れる場合，図 3 (2) は予測を無視し，通常の DnD を行う場合にそれぞれ対応している．

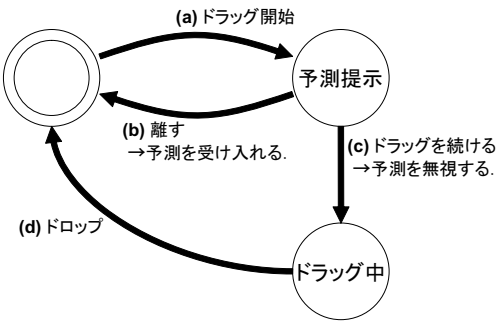

図 2. Drag-and-guess の状態遷移

Drag-and-guess は通常の DnD と同様，オブジェクトのドラッグから始まる（図 2 および図 3 の (a)）．

ドラッグ開始をきっかけとしてシステムは予測提示状態となり，予測される振り分け先とカーソルとを結ぶ形で予測を提示する（図 3 の上から二段目）．予測される振り分け先が可視でない場合は，ツリーの展開とコンポーネントのスクロールを行う．この可視化は一時的で，操作が終了すると操作前の状態に戻る．

予測提示状態で提示された予測を受け入れる場合には，マウスボタンを離すことで振り分けが完了する．それに対し，予測を無視する場合にはそのままドラッグを続ける．カーソルを囲む円形領域を出たとき（図 2 および図 3 の (c)）に予測提示はキャンセルされ，通常の DnD となる．一時的に展開されていたツリーやスクロールは元の状態に戻る．

DnD 中にツリー上の収納されたフォルダ上で一定時間待機するとフォルダが展開されるので，予測が外れた場合にも時間をかければ目的のフォルダに DnD を行うことができる．

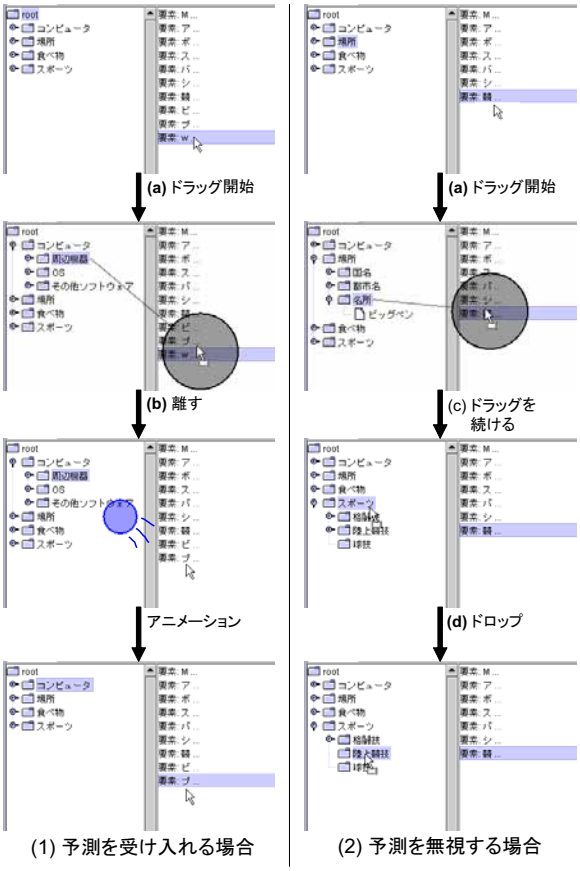

図 3. Drag-and-guess 使用の様子

3.3 利点

Drag-and-guess は DnD を以下の点で改善する．

- ドロップ作業が省略されることで，小さな目標を狙う操作が減る．
- 目標を見える状態にする作業が減る．

また，drag-and-guess はボタンなどの単純な予測提示手法と比較して以下に挙げる点で優れている．

- 予測を提示するために追加の widget やスペースを必要としない．
- 振り分け時にフォルダの階層構造を意識することが状況の理解を助ける．
- 予測が正しくない場合，DnD へなめらかに移行することができる．

3.4 予測

現プロトタイプでは予測はあらかじめ用意した正解に基づいて行うようになっている．予測精度の変

化と提案手法の有効性の変化の対応を観察するため，予測精度は自由に変えられるようにした．

4　評価実験

4.1　実験方法

実装したプロトタイプを用いて drag-and-guess (DnG) と予測をひとつのボタンで提示する手法 (Btn) を比較する実験を行った．

8 人の被験者にテストデータの振り分けタスクを行ってもらった．すべての被験者が DnG を用いて 5 ブロック，Btn を用いて 5 ブロックのタスクを行った．各ブロックは 20 回の振り分けから成り，異なる予測精度に設定された (10%, 30%, 50%, 70%, 90%)．DnG のブロックと Btn のブロックは交互に配置された．また，被験者の半数が DnG を先に，残りの半数は Btn を先に用いた．

被験者がリスト中のオブジェクトを選択してから，実際にそのオブジェクトが振り分けられるまでの時間を測定した．

4.2　実験結果

実験結果を図 4 に示す．分散分析の結果，「予測精度」と「振り分け手法」の主効果がともに有意であった ($p<0.001$)．交互作用は有意ではなかった．さらに t 検定の結果，DnG は予測精度が 30%, 50%のとき有意に速く ($p<0.05$)，その他の予測精度では有意な差はなかった．これは，提案手法を用いた場合予測が誤っていると被験者が判断したときに DnD に切り替えるのが容易であったからではないかと考えられる．

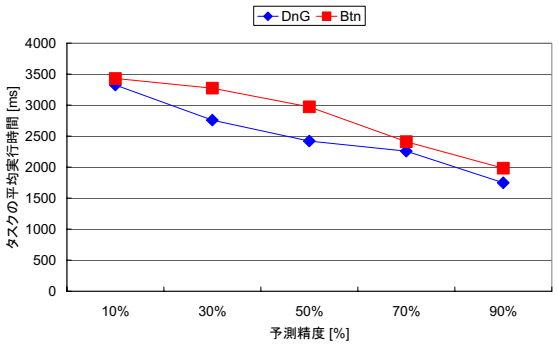

図 4．タスク平均実行時間と予測精度の関係

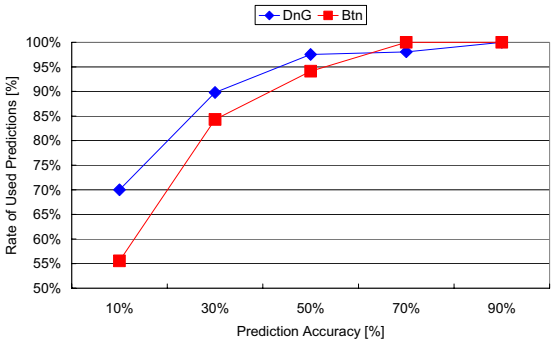

図 5．使用された正しい予測の割合

図 5 は使用された正しい予測の割合である．予測精度が低下すると予測が使われなくなるという全体的な傾向に加え，DnG による提示の方がより使われたという傾向を見て取ることができる．

5　議論

5.1　予測提示方法

まず，予測を表示する位置としてプロトタイプで採用した方法のほかに二つの候補について検討を行った（図 6）．予備的な実験と議論の結果，ドラッグ開始のタイミングにおいて多くの人がドロップ目標アイコンの付近を見ていたため，本稿で説明した表示位置を採用した．

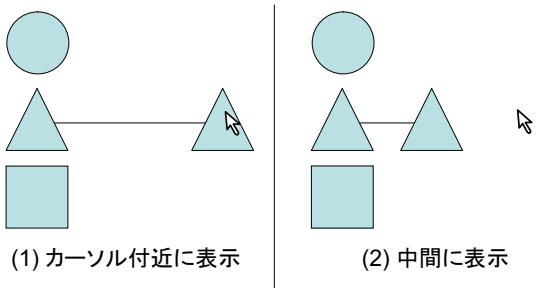

図 6．予測表示位置の候補

予測を提示するタイミングによってはほかの方法も考慮に入れる必要がある．例えば，ドラッグするアイコンをつかんだときに予測を提示する場合は，カーソル付近が適していると考えられる (図 6 (1))．

また，目標アイコンから離れた位置に予測を提示する場合には，目標アイコン周囲の構造をどれだけ表示するかという問題が生じる（図 7）．周囲の構造を示すことは予測の正否を判断する助けとなり得るが，より多くの表示領域を必要とする．本稿のプロトタイプでは，実際の振り分け先を展開する形で予測提示を行ったため，自然な形で周囲の構造を示すことができた．

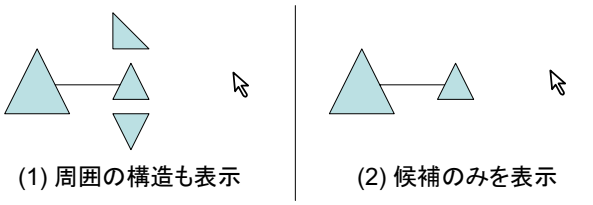

図 7. 周辺構造の表示

5.2 ドラッグ方向と距離

Drag-and-guess でも drag-and-pop[3] やアイコン投げ[7] のように，ドラッグ方向を用いることができる（図 8）．ドラッグ方向の先にあるアイコンを優先的に提示することで，静的な予測精度の不足を補うことができる．

また，より遠くにあるアイコンを優先的に提示することで DnD の平均的な負担を減らすことができると考えられる．

メールの振り分け作業においては，ドラッグ方向も距離もほぼ一定であるため，シンプルさを重視した本稿のプロトタイプではこの手法は採用しなかった．提案手法をデスクトップ環境に適用する場合には，静的な予測に以上のような修正を加えることが有効である．

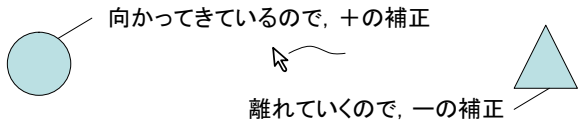

図 8. ドラッグ方向による予測の補正

5.3 複数候補の提示

MailCat では，上位３つの候補を提示することで予測精度の不足を補うことができることが示された[4]．提案手法でも同様に複数候補を提示することが考えられる．その場合，２位以下の候補をどのように提示・選択するかが問題となる．図 9 にその一例を与える．