D-13-2

モバイルエージェントフレームワークMiLogにおけるトレイルスタックの最適化手法について

A TrailStack Optimization Method on the Logic-based Mobile Agent Framework MiLog

福田直樹　　新谷虎松
Naoki FUKUTA　Toramatsu SHINTANI
名古屋工業大学 知能情報システム学科
Dept. of Intelligent Computer Systems, Nagoya Institute of Technology.

1. はじめに

モバイルエージェントは，移動の際にそのプログラムの実行に関する情報を保存し，移動先でその情報を復元し，実行を再開する．モビリティのクラスは，移動時に保存される情報の性質の違いから，Weak MigrationとStrong Migrationの2つのクラスに分類できる．Strong Migrationでは，実行状態とデータ領域の両方が移動される．移動後のプログラムの実行は，移動前とまったく同一の位置から再開される．Weak Migrationでは，実行状態は保存されず，データ領域のみが保存される．プログラムは，移動前に実行状態をデータ領域へ保存するための手続きを行い，移動後にデータ領域から実行状態を復元するための手続きを行う必要がある．Strong Migrationでは，プログラム中に実行状態のデータ領域への保存および復元手続きを記述する必要がないため，プログラムが記述しやすく理解しやすいという利点がある．Strong Migrationでは，移動前のプログラムの実行状態を取り込むための機構が必要となる．Strong Mitrationの実装においては，プログラムの実行状態が肥大化した場合に，マイグレーションのオーバヘッドが大きくなる点が問題となる．本論文では，Strong Migrationを論理型言語処理系に適用するための，最適化手法について述べる．

2. トレイルスタックの最適化

論理型言語処理系において，実行状態を構成するスタックの中で，実行時における肥大化が特に顕著なのが，トレイルスタックである．トレイルスタックとは，節の実行失敗時に変数束縛を解除するためのデータを保存しておくためのスタックである．トレイルスタックの古典的な最適化手法として，変数の生成を抑制する手法があるが，生成されてしまった変数に対する最適化手法については，あまり議論されていない．本論文においては，定理1に基づくトレイルスタックの最適化を行う．

定理 1 (束縛解除が不要な変数の十分条件) *Box* モデルにおける *Box B* の識別番号を bi_B とする．任意の *Box* B_i, B_j において，$birth(B_i) < birth(B_j) \rightarrow bi_{B_i} < bi_{B_j}$ とする．ここで，$birth(B_n)$ は，*Box* B_n が生成された時刻を意味する．*Box B* に *Head* 部分が単一化された *Horn* 節 を H_B とする．H_B において，同一の名前を持つ変数は唯一の変数として生成されるものとする．H_B において，変数 X が生成されたときの識別番号を $ID_X = bi_B$ のように決める．ある *Box* B_n において，変数 X に何らかの値が束縛されたとき，束縛解除情報 $trail(X)$ は $[X, bi_B, bi_{B_n}]$ と表現される．選択点スタックの先頭にある *Box* を $B_{CP_{top}}$ とすると，$bi_B > bi_{B_{CP_{top}}}$ なる変数 X は束縛解除が不要である．

3. 評価

本手法を，本研究室で開発したモバイルエージェント記述フレームワークMiLog[1][1]に適用し，*BiddingBot*[2]システムにおいてその効果を測定した．対象としたプログラムは，特定のWWWサイトから情報を抽出するタスクを行う．本プログラムは，我々が提案しているオークション支援システム *BiddingBot* において，WWWサイトから財の価格や入札期限などの情報を取得するために用いられている．実験は，PentiumII 300MHzのCPUを搭載したノートパソコン上において，JAVA2 SDK 1.2.2を用いて行った．実験では，プログラム実行中におけるマイグレーションに必要なデータ量について，本最適化を適用した場合と適用しない場合を比較した．マイグレーション時には，プログラム p とスタック s の両方が転送されるため，スタックが空の場合も測定し，差分をとった．実験結果を，表1に示す．本最適化を適用した際に，s の値を 38.0% 削減することができた．

表 1: トレイルスタック最適化の効果

	最適化あり	最適化なし
p + s [bytes]	58899	85962
p [bytes]	15159	15351
s [bytes]	43740	70611

4. おわりに

本論文では，論理型言語におけるメモリ使用量の最小化手法として，トレイルスタックの最適化手法に着目した．トレイルスタックの最適化手法をMiLog上に実装した．BiddingBotシステムにおいて評価実験を行い，マイグレーションに必要なデータ量を削減できた．

参考文献

[1] 福田直樹，伊藤孝行，新谷虎松：“Weblog: WWWにおける情報収集エージェントのための論理型記述言語の実現,” 第8回マルチエージェントと協調ワークショップ (MACC99)，1999.

[2] T. Ito, N. Fukuta, T. Shintani and K. Sycara : ”BiddingBot: A Multiagent Support System for Cooperative Bidding in Multiple Auctions,” In Proc. of ICMAS 2000(to appear), 2000.

[1] Weblogは MiLogの古い名称である．