6N-3

PIMOSのトレーサ

中尾浩一[1]、近山隆[2]、中島克人[2]、大西諭[2]
1: 応用技術(株) 2:(財)新世代コンピュータ技術開発機構

1 はじめに

第五世代コンピュータの開発プロジェクトでは、並列推論マシン用のオペレーティング・システム PIMOS (Parallel Inference Machine Operating System)[1][2] の開発を行っている。PIMOS におけるユーザ・プログラムのデバッグ機能として現在トレーサやインスペクタを開発中である。

本稿ではトレーサの概要とその実現方式について報告する。

2 目的

プログラムのデバッグのために実行をトレースする機能で重要なことは、必要な部分以外のトレースを抑制し本来のデバッグ対象に注意を集中できるようにすることである。

通常の逐次型言語では、プログラム中の一箇所から起動されたプログラム部分が実行を終えるまでは他の部分の実行が開始されることはなく、実行中のある適当な時間間隔についてその中の実行の全てをトレースすれば、ある箇所から起動されたプログラムのみに限定したトレースを行うことができる。

一方、並列言語である KL1 のプログラムでは、概念上ゴールは全て並列に実行される。実際にも、実行は必要なデータが揃ったゴールから行われるので、実行の時間順序はゴール間の論理的な呼びだし関係と直接関係しない。このため、時間間隔を限ったトレース方式では望むようなトレースの抑制を実現することはできない。また、KL1 プログラムの実行は、一般にストリームを介してお互いに通信をしあうプロセスの集まりとして捉えられる。各プロセスの振舞いは時間の推移とは無関係であり、デバッグする者にとっては、メッセージのやり取りによってプロセスの状態が変わっていく過程を見ることが必要となる。

そこで PIMOS ではゴールの親子関係だけに注目して実行を追っていくトレース方式を実現している。この方式では、呼びだし関係の木構造の中でトレースが不要なゴールの枝に対してはトレースを抑制して実行を進めていく。トレースの対象は"リダクション"(ゴールをガード部の実行が成功するプログラム節の適用により、ボディに指定されたサブゴールに置き換えること)である。リダクションは KL1 の言語レベルで最小の単位であり、それよりも細かいレベルの実行は処理系のインプリメンテーションに依存する。したがって、トレースの実行はひとつのリダクションの終了ごとに、その結果生じたサブゴールを報告する形で進められていく。

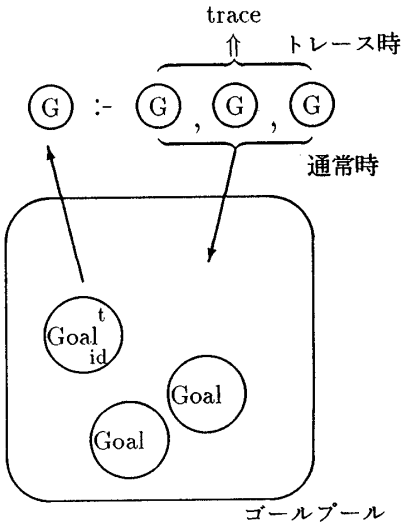

図 1: 処理系の実行イメージ

3 トレース機構

処理系における KL1 の実行では、通常はゴールのリダクションの結果生じたサブゴールは、プロセサごとに持っているプール(ゴールプール)に保持され、後にプールから取り出され実行される。一方、トレースを指定されたゴールのリダクションでは、サブゴールはプールには格納されず、リデュースしたゴールの識別子と共にトレーサに報告される(図 1)。

この識別子はゴールのトレースを指定する際に与えるものであり、これを管理しておくことでどのゴールのリダクションであるのかを知ることができる。識別子のやり取りによってトレースを実現しているところがこのトレーサの特徴といえる。トレーサは任意のサブゴールに対してトレースを指定し、サブゴールはゴールプールに格納される。トレースが指定されたサブゴールには与えられた識別子と、プールからの取り出し時にそれと判るようにトレースの印が付けられる。

トレースが抑制されているゴールは通常となんら変わりなく実行されるので、トレースによる実行速度の低下は最小限に抑えられる。

Tracer on PIMOS
Koichi NAKAO[1], Takashi CHIKAYAMA[2], Katsuto NAKAJIMA[2], Satoshi ONISHI[2]
1: Applied Technology Co.,Ltd. 2: Institute for New Generation Computer Technology

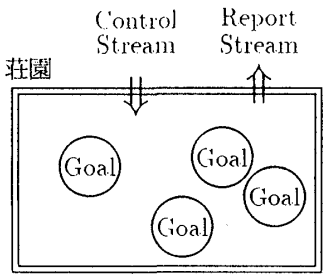

図 2: 荘園の概念図

3.1 荘園とレポート・ストリーム

トレースの実現には KL1 の荘園の機能を利用している. 荘園とは KL1 プログラムの実行や資源の管理を行うための基本機能であり, コントロール・ストリームとレポート・ストリームを介して, その内部のゴール群の実行を制御 / 観測できる実行管理単位である (図 2).

コントロール・ストリームには実行の中断, 再開, 放棄などのメッセージを送ることができ, レポート・ストリームからは, 実行の終了, 例外事象の発生, デッドロック・ゴールの発見などが報告される.

トレースが指定されたゴールのリダクションもこのレポートストリームを通してトレーサに報告される.

3.2 トレース報告

トレースが指定されたゴールのリダクションは次の形式で報告される.

exception(TraceID, SubGoalsInfoList, NewGoal)

TraceID: トレース要因識別子. トレース対象を指定した際に与えた識別子. リデュースしたゴールを表す. 識別子には任意のデータを与えることができる.

SubGoalsInfoList: リダクションの結果生じたサブゴールの情報のリスト. リストの各要素はサブゴールのタイプ, 呼びだし先コードと引数ベクタからなる. サブゴールには, ユーザ定義述語の他にボディ組込述語やボディ・ユニフィケーションも含まれる.

NewGoal: 新ゴール. トレース報告元で実行すべきゴールのコードと引数ベクタを指定する変数. 報告されたサブゴールはこの変数を介して実行される.

3.3 対象ゴールの指定方法

トレースの指定には通常の高階述語 (apply) の代わりに, トレース用の高階述語を用いる.

apply_tracing(Code,ArgV,TraceID)

Code: トレース指定されたゴールの呼びだし先コード.

ArgV: 引数ベクタ.

TraceID: トレース要因識別子.

この述語によって起動されたゴールのリダクションをトレース対象とする. トレーサはサブゴールのトレースの指定にしたがって, トレース報告中の NewGoal に apply_tracing を使うか apply を使うかを決定する.

4 トレーサの実装と機能

トレーサを実現するにあたり, トレース要因識別子には整数を用い, 識別子をキーとしてゴール情報 (ゴールのコードと引数のベクタ) をキー付きプールで管理した. トレーサの主な機能を紹介する.

トレース出力: リダクションの報告にはゴールの引数の状態だけではなく, ゴールをリデュースしたプロセサの番号や, 優先度, サブゴールのプラグマ指定 (プロセサ指定, 優先度指定) なども出力される.

スパイ: スパイの指定に合致するゴールをリデュースしたときにトレースするものと, スパイの指定に合致するゴールをサブゴールとして持つゴールをリデュースしたときにトレースするものの二種類のスパイ機能を用意した.

インスペクタ: ゴールやサブゴールの引数の状態を調べることができる.

トレース指定ゴールリスト: トレースを指定して実行したゴールのうち, トレース報告がなされていないゴールを一覧表示する機能. デッドロックするようなゴールは, トレースを指定して実行していれば必ずこのゴール群の中に含まれている.

5 今後の開発予定

今後はユーザ・インタフェースの改良と共に機能の拡張を行っていく.

現在, トレース対象の指定はゴールに対してのみであり, 制御の流れに注目したトレース方式となっている. これとは別に, ユニフィケーションの際に変数が具体化されたことによって可能になるリダクションをトレース対象とする方式も検討中である. トレースを指定した変数が具体化される場合, ガード中でその変数の値を読むようなリダクションはトレースの対象となる. これは, データの流れに注目したトレース指定を目的としたものであり, より具体的にはユニフィケーションとリダクションの因果関係に注目したトレース指定方式となる.

デッドロックの検出機能は近々搭載する予定である.

アルゴリズミックなデバッグ方式 [3][4] についてもその有効性と実現方式について検討中であるが, 実行時間やメモリ量といった現実的な制限を考えると実現は難しい.

参考文献

[1] T. Chikayama et al. : *Overview of the Parallel Inference Machine Operating System (PIMOS)*, In Proc. of FGCS'88, Vol. 1, pp. 230-251, 1988.

[2] 佐藤 他 : PIMOS の資源管理方式, 並列処理シンポジウム JPSS'89, S4-1, 1989-2.

[3] E. Shapiro. *Algorithmic Program Debugging*, MIT Press, 1983.

[4] A. Takeuchi, *Algorithmic Debugging of GHC Programs and its implementation in GHC*, Technical Report TR-185, ICOT, 1986.