Code Gear、 Data Gear に基づく OS のプロトタイプ

伊 波 立 樹[†1]　東恩納　琢偉[†2]　河　野　真　治[†2]

当研究室では 処理の単位を Code Gear、 データの単位を Data Gear を用いて並列実行を行う Gears OS を開発している Gears OS では 並列実行をするための Task を Code Gear と Data Gear の組で表現する。Task の依存関係は Code Gear を実行するために必要な Input Data Gear と Code Gear で作られる Output Data Gear によって決定し、それにそって並列実行を行う。依存関係の解決などの Meta Computation の実行は Meta Code Gear で行われる。Meta Code Gear は Code Gear に対応しており、 Code Gear が実行した後にそれに対応した Meta Code Gear が実行される。本論文では Gears OS のプロトタイプとして並列処理機構を設計し、 CbC(Continuation based C) で実装する。

Tatsuki IHA ,[†1] Takui HIGASHIONNA [†2] and Shinji KONO [†2]

1. GearsOS

2. Code Gear と Data Gear

Gears OS はプログラムの単位として Gear を用いる。Gear は並列実行の単位、データの分割、Gear 間の接続等になる。

Code Gear はプログラムの処理そのものである。Code Gear は任意の数の Input Data Gear を参照し、処理が完了すると任意の数の Output Data Gear に書き込む。Code Gear は接続された Data Gear 以外には参照行わない。

Data Gear は Data そのものを表しており、int や文字列などの Primitive Data Type が入っている。

Gears OS では Code Gear と Input / Output Data Gear の対応から依存関係を解決し、Code Gear の並列実行を可能とする。図?? に Code Gear と Data Gear を使用した実行を示す。

Gear の特徴として処理やデータの構造が Code Gear、 Data Gear に閉じていることにある。これにより、実行時間、メモリ使用量などを予想可能なものにする事が可能になる。

図 1　Code Gear と Data Gear での実行

3. Continuation based C

Gears OS の実装は本研究室で開発している CbC(Continuation based C) を用いて行う。CbC は処理を Code Segment を用いて記述することを基本としているため、 Gears OS の Code Gear を記述するのに適している。

CbC のプログラムでは C の関数の代わりに Code Segment を用いてい処理を記述している。Code Segment は C の関数と異なり戻り値を持たない。Code

†1 琉球大学大学院理工学研究科情報工学専攻
Interdisciplinary Information Engineering, Graduate School of Engineering and Science, University of the Ryukyus.
†2 琉球大学工学部情報工学科
Information Engineering, University of the Ryukyus.

Segment の宣言は C の関数の構文と同様に行い、型に __code を使うことで宣言できる。

Code Segment から Code Segment への移動は goto の後に Code Segment 名と引数を並べた記述するという構文を用いて行う。この goto による処理の遷移を継続と呼ぶ。図 1 は Code Segment 間の継続関係を表している。

C では関数呼び出しを行うたび、関数の引数の値がスタックに積まれていくが、Code Segment では戻り値を持たないため、スタックに値を積んでいく必要がなくスタックを変更する必要が無い。このようなスタックに値を積まない継続、つまり呼び出し元の環境を持たない継続を軽量継続と呼ぶ。軽量継続により、並列化、ループ制御、関数コールとスタックの操作を意識した最適化がソースコードレベルで行えるようにする。

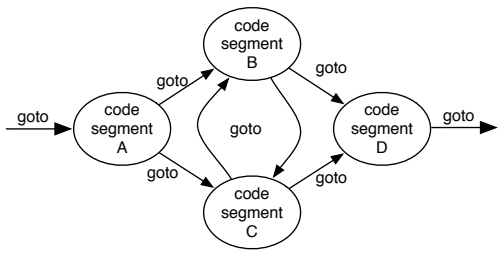

図 2 goto による Code Segment 間の接続

4. CbC での Gears OS の構文サポート

CbC は Gears OS の構文のサポートを行う。

Gesrs OS では Context という接続可能な Data Gear のリストからデータを取り出して処理行う。しかし、 Context を直接扱うのはセキュリティ上好ましくない。そこで Gears OS では Context から必要なデータを取り出して Code Gear に接続する stub を定義する。stub は Code Gear から推論することが可能のため、 CbC は自動的に stub の生成を行う。

また、Code Gear の遷移には meta computation を行うために Meta Code Gear を挟む。CbC では Meta Code Gear への接続も自動的に行うようにする。

5. ま と め

参 考 文 献

1) 宮國 渡，河野真治，神里 晃，杉山千秋：Cell 用の Fine-grain Task Manager の実装，情報処理学会システムソフトウェアとオペレーティング・システム研究会 (OS) (2008).
2) 赤嶺一樹，河野真治：DataSegment API を用いた分散フレームワークの設計，日本ソフトウェア科学会第 28 回大会論文集 (2011).
3) Sony Corporation: Cell broadband engine architecture (2005).
4) 河野真治，杉本 優：Code Segment と Data Segment によるプログラミング手法，第 54 回プログラミング・シンポジウム (2013).
5) 河野真治，島袋 仁：C with Continuation と、その PlayStation への応用，情報処理学会システムソフトウェアとオペレーティング・システム研究会 (OS) (2000).
6) 徳森海斗，河野真治：Continuation based C の LLVM/clang 3.5 上の実装について，情報処理学会システムソフトウェアとオペレーティング・システム研究会 (OS) (2014).
7) Moggi, E.: Computational lambda-calculus and monads, *Proceedings of the Fourth Annual Symposium on Logic in computer science* (1989).
8) 下地篤樹，河野真治：線形時相論理による Continuation based C プログラムの検証，情報処理学会システムソフトウェアとオペレーティング・システム研究会 (OS) (2007).
9) Aaftab Munshi, Khronos OpenCL Working Group: *The OpenCL Specification Version 1.0* (2007).
10) : CUDA, https://developer.nvidia.com/category/zone/cuda-zone/.
11) : MessagePack, http://msgpack.org/.