コンパイラにおける生存区間重複検査の高速化に関する一方式

3J－4

田中 旭　佐山 旬子　湯川 博司　小谷 謙介
松下電子工業（株）

1. 背景

我々は、高経済性と高性能性の両立を追及した組み込みマイコンMN10200シリーズの開発を行っている。

MN10200マイコンは、アドレスレジスタ、データレジスタ、各４本のレジスタ構成を持っている。この構成の利点は、マイコンコア自体のコスト削減が図かれることと、組み込みプログラムが格納されるROMサイズの減少が可能となることである。これは、機械語命令のレジスタ指定フィールドが２ビットで済むので、コンパイラが生成する頻度の高いレジスタ間演算や転送命令等の基本命令を１バイト化することが可能となり、結果的に組み込みプログラムのサイズを小さく抑えることが可能となるからである。[1]

しかし、コンパイラにおける変数へのレジスタ割付け処理を工夫しないと、汎用レジスタ構成に比べてかえってレジスタ間やレジスタメモリ間の転送命令が多数発生し上記の利点を活かせなくなる可能性がある。

そこで、文献[2]において、MN10200のような少数で使用制限のあるレジスタ構成をもつマイコンの、レジスタ割付け方法の一方式について提案を行った。この方式のように、少数のレジスタに効率よく変数を割付けるためには、変数の値が有効となるプログラム部分である生存区間を精密に求め、かつ変数の使用状況を詳細に調べる必要がある。よって変数の数が多い場合、レジスタ割付け処理の時間も増大する。

本稿では、コンパイラのレジスタ割付け処理において必須でありかつ比較的時間を要する、生存区間の重なり検出処理の時間短縮に関して報告する。

A Method of Reducing Time as the Live Range Interfering Check Process in the Compiler
Akira TANAKA, Junko SAYAMA, Hiroshi YUKAWA, Kensuke ODANI
Matsushita Electronics Corporation

2. レジスタ割付けにおける問題点

レジスタ割付けの原則は、生存区間が重なる変数の間では異なるレジスタを割付けることである。そのため、全ての変数に関して生存区間が重なる変数を検出することが必要であり、通常は変数の数がN個であれば、N(N-1)/2 回の重複検査が必要となる。 また一般的に、生存区間は、コンパイラで用いる内部表現である中間命令の集合として表現される。よって、生存区間の重複検査は中間命令集合の論理積で行われる。

表１は、サンプルプログラムのサイズ(SIZE)、１関数における変数の個数うち最大の数(N)、生存区間の重複検査時間(IT)、全コンパイル時間(CT)を示すものである。サンプル１はベンチマークプログラム、サンプル２、３はＣ言語コンパイラの検証用プログラム、サンプル４、５は、組み込み用プログラムである。

表１で特に問題となっているのは、コンパイラ自体のテストに使用される検証用プログラムのコンパイルに数時間も要することであり、コンパイラ開発期間の短縮化の大きな障害となる。また、今回の例ではそれほど顕著ではないが、年々組み込みプログラムの規模は増大してきており、その変数の数も増大することが考えられ、

表1. サンプルプログラムの評価値

	SIZE	N	IT	CT	IT/CT
sample1	1,048	179	0.8	14.4	0.056
sample2	9,276	960	48.0	128.2	0.374
sample3	65,771	5,263	12,848.9	16,948.4	0.758
sample4	25,725	1,673	153.5	511.0	0.300
sample5	25,667	469	13.0	158.2	0.082

SIZE:line, N:number, CT:sec., IT:sec.

重複検査の増大は、組み込みプログラム開発期間の短縮化の阻害要因になる可能性もある。

よって、全ての変数の使用状況を詳細に調べ、より効率のよいレジスタ割付けを行うには、その割付け処理方式も重要であるが、必須である生存区間の重複検査処理の実行時間を短縮することも解決しなくてはならない。

3. 解決策

3.1 単純ブロック分割

上述の問題点を解決するために我々は、変数への値の代入(変数の定義)の後、1回のみしか使用されないような、コンパイラが内部で生成する一時変数が多数存在することに着目して、まず基本ブロックを次の性質を持った単純ブロックに分割した。

単純ブロック:

単純ブロックの最後の中間命令で定義される変数を除いて、単純ブロック内で定義される変数は、その使用が同じ単純ブロック内に存在する。

図1は基本ブロックを単純ブロックに分割する一つのアルゴリズムである。このアルゴリズムは、単に分割を行うだけでなく、中間命令の移動により、より大きな単純ブロックを形成するようになっている。

図1. 単純ブロック生成アルゴリズム

s1. 基本ブロック内の全ての中間命令を一つだけ含む単純ブロックを生成する。

s2. 実行順に以下の処理を行っていない単純ブロックb1を取り出す。

s3. b1が基本ブロックの最後のとき処理を終了。

s4. b1の定義s1で定義される変数vの使用が1回であり、その使用がb1に存在するとき次のステップに進み、そうでない時ステップs2を行う。

s5. vの使用までにb1で使用されている変数の定義があるときステップs2を行い、そうでないとき次のステップへ進む。

s6. vを使用する中間命令を含む単純ブロックb2を取り出す。

s7. b2にb1が含む全ての中間命令を移動させ、ステップs2へ戻る。

3.2 ブロック内生存変数に着目した生存区間重複検査方式

上述の単純ブロック生成の後に、変数の生存区間検出の際に、変数の生存区間の存在範囲によって変数を次の3つに分類し、分類から明らかに生存区間が重ならない変数間での生存区間の重複検査処理を省くことによって、その処理の短縮化を図った。

分類:

(a)単純ブロック内のみに生存区間が存在する変数
(b)基本ブロック内のみに生存区間が存在するが、複数の単純ブロックに跨って生存区間が存在する変数
(c)複数の基本ブロックに跨って生存区間が存在する変数

具体的には、まず(a)に分類される変数に関しては、単純ブロック毎に変数の集合を設け、変数の生存区間検出の際に、その変数の生存区間が存在する単純ブロックの集合に変数を格納し、生存区間の重複検査処理の際に、異なる単純ブロックの変数集合に属する変数間での、生存区間の重複検査を省略する。

同様に、(b)に分類される変数に関しては、基本ブロック毎に変数の集合を設け、変数の生存区間検出の際に、その変数の生存区間が存在する基本ブロックの集合に変数を格納し置き、生存区間の重複検出処理の際に、異なる基本ブロックの変数集合に属する変数間での、生存区間の重複検査を省略する。

4. 評価

表2は本方式による処理時間である。ITの改善率(IMP)は、表1と比較して約60%〜90%であり、良好な結果が得られた。

表2. 本方式の評価値

	IT	CT	IT/CT	IMP
sample1	0.3	14.1	0.022	61.7
sample2	10.1	92.3	0.109	79.0
sample3	2,929.6	7149.4	0.409	77.2
sample4	15.5	376.4	0.041	89.9
sample5	2.2	151.7	0.014	83.3

CT, IT:sec. IMP:%

5. 結び

本文では、コンパイラにおけるレジスタ割付けにおいて、最も時間を費す生存区間の重複検査処理の高速化について述べた。

今後は、単純ブロックの大きさと、重複検査処理時間とのトレードオフに関して評価を行い、本単純ブロック生成アルゴリズムの改善を行う予定である。

【参考文献】

[1]桧垣他、" 機器組込み用小型16ビットマイコンMN10200 のパイプライン制御方式" 電情信秋季大会、D-57, 1992

[2]田中他、" 組込みマイコン用最適化コンパイラにおけるレジスタ割付けの一方式" 情処春季大会、6G-05, Mar. 1994